A Benders decomposition method for locating stations in a one-way electric car sharing system under demand uncertainty

Hatice Calik, Bernard Fortz

A Benders decomposition method for locating stations in a one-way electric car sharing system under demand uncertainty

Hatice Çalık[a,b,*], Bernard Fortz[a,c]

[a] *Department of Computer Science, Université libre de Bruxelles, 1050 Brussels, Belgium*
[b] *Université de Lorraine, CNRS, LORIA, F-54000 Nancy, France*
[c] *INOCS, INRIA Lille Nord-Europe, France*

Abstract

We focus on a problem of locating recharging stations in one-way station based electric car sharing systems which operate under demand uncertainty. We model this problem as a mixed integer stochastic program and develop a Benders decomposition algorithm based on this formulation. We integrate a stabilization procedure to our algorithm and conduct a large-scale experimental study on our methods. To conduct the computational experiments, we develop a demand forecasting method allowing to generate many demand scenarios. The method is applied to real data from Manhattan taxi trips. We are able to solve problems with 100 to 500 scenarios, each scenario including 1000 to 5000 individual customer requests, under high and low cost values and 5 to 15 mins of accessibility restrictions, which is measured as the maximum walking time to the operating stations.

Keywords: Location, Urban Mobility, Electric Car Sharing, Benders Decomposition, Mixed Integer Stochastic Programming, Stochastic Demand

1. Introduction

The increasing number of privately owned cars causes significant amount of pollution, traffic congestion, and parking problems in urban cities. The private cars are usually parked for very long hours and their utilization rates are very low. The car sharing systems are based on shared usage of vehicles by multiple people and the utilization rate of these vehicles, usually owned by companies or organizations, are much higher compared to the private ones. They usually serve their users based on a subscription system and charge each usage by traveling distance or time. Due to this kind of pricing, the users tend to drive less and therefore, create less traffic. Together with the integration of environmental friendly vehicles, car sharing systems have a high potential of reducing these crucial urban problems.

The car sharing systems can be grouped into two categories as station based and free floating depending on whether the customers are required to visit certain stations or not. In station based systems, where the customers need to visit stations to pick-up and drop the cars, two types of trips can be observed: i) round trips which force to leave the cars to the stations that they were taken from and ii) one-way trips where the cars can be dropped at stations different than the originating one. One-way systems provide more flexibility

*Corresponding Author.
Email addresses: hatice.calik@ulb.ac.be (Hatice Çalık), bernard.fortz@ulb.ac.be (Bernard Fortz)

to the users but requires a more sophisticated management as otherwise there might be significant load imbalances at stations. In free-floating systems, the cars can be parked anywhere within a pre-defined zone and customers can pick the cars from wherever they are parked. Although they are more attractive from a flexibility perspective, free floating systems do not guarantee an available parking space to the users at the end of their trips and the cars need to be parked with at least a certain level of fuel or battery. Moreover, these systems in general require relocation of cars in order to balance the availability at different parts of the city and to perform refueling/recharging operations.

In this work, we focus on design of a one way station based electric car sharing (ECS) system that operates under demand uncertainty. We assume that the users will be served with a fleet of identical electric cars. Our main goal is to decide on the location and number of stations with limited capacities and the initial level of cars at each station to operate the system in order to maximize the expected profit of the operating company. The expected profit is equal to the expected revenue obtained from served customers minus the fixed cost of opening stations and purchase cost of cars. The electric cars have a shorter range than the conventional cars and the recharging operations take much longer. Therefore, the location of stations plays a crucial role in service quality. To this end, we develop exact methods to design a system that takes into account accessibility of users to located stations, availability of a car at the departure station, and availability of an empty spot at the arrival station for forecasted customer requests. We measure accessibility by the walking time from origin and destination points, in other words, a station is accessible by a customer if their origin and destination points are within a certain walking time to some operating stations.

We consider general capacities (non-identical) for stations and assume that the number of charging units is equal to the number of parking spots at the stations. The cars are plugged as soon as they are parked to a station and they are recharged until full battery capacity before being available for the next customer. This assumption enables solving real size problems with strategical level decisions [10, 14].

We aim to solve large scale problems of locating recharging stations without aggregation of demand. In order to represent uncertainty in demand, we consider multiple scenarios with occurrence probability. Each scenario consists of a certain number of customer requests and each request is associated with an origin node, a destination node, and a starting time. The problem under consideration requires decisions on the number and location of stations, the number of cars available at each station, the customers to be served, the stations they need to visit, and the time of each visit. We assume that relocations will be be done outside the planning period (e.g. overnight). Therefore, we do not consider other operational decisions such as relocation time/network of cars and number of operators to hire for relocation etc.

We formulate this problem as a two stage stochastic programming model and develop a Benders decomposition algorithm based on its deterministic equivalent. In order to test our methods, we develop a demand forecasting method that uses historical data to generate new requests. We evaluate the performance of our algorithm under different parametric and structural settings by using problem instances obtained from a real taxi trip data of Manhattan. From the experimental studies we conducted on the formulation, we observe that the average linear programming relaxation (LP) gap of the formulation is less than 0.4% on the instances tested. Moreover, a partial relaxation of the formulation provides the same objective value with the original model for most problems with an average gap less than 0.01% in our test bed. We further strengthen our Benders method with a stabilization procedure, which improves the average performance of our algorithm, especially, on instances with larger number of scenarios.

The rest of this paper is organized as follows: In Section 2, we present a brief summary of the related works in the literature. In Section 3, we provide a mathematical description of our problem with some notation and provide the details of our mixed integer programming formulation. Section 4 presents the results of the computational study we conducted on our formulation and its relaxations. We dedicate Section 5 to a brief summary of the classical Benders decomposition method, its adaptation to pure integer programming (IP) problems in the literature, and a detailed description of the Benders algorithm that we propose. Section 6 follows with the computational study on our Benders algorithm and a stabilized version of it. We conclude in Section 7 with a summary of our results and perspectives on future research.

2. Related work

The current studies in the car sharing literature mostly focus on tactical and operational level decision problems. Tactical and operational level problems concerning electric vehicles already attracted attention in logistics systems and several studies in the literature focused on different variants of electric vehicle routing problems [26, 28, 45, 43]. Within the context of car sharing systems, the most commonly considered problem is the relocation of cars to increase availability in high demand areas and avoid surplus of cars at the low demand points [4, 15, 19, 22, 35, 38, 39, 46, 33, 11]. On the other hand, as also indicated in a recent review [12], the car sharing literature on strategic level decision problems involving location of stations is relatively sparse.

Optimal location of stations in one-way car sharing systems is first addressed by Correia and Antunes [24]. The authors propose a mixed integer programming formulation that maximizes the profit of the operator by considering several cost and revenue factors. They compare the model, which enforces service via closest stations to origins and destinations, under three service strategies: (i) The operator has the possibility to decide on which customer requests to serve without any restriction (ii) all requests have to be served (iii) the operator can reject a request only if there is no car available at the starting station. The experiments on a real data from Lisbon, Portugal, show that serving all customers might decrease the profit significantly. Later, Correia et al. [25] extend the formulations of Correia and Antunes [24] to a more flexible system where the assumption of service through closest stations is relaxed. The experiments in the latter study reveal that this kind of flexibility increased the profit of the operator together with the introduction of vehicle stock information.

In addition to the optimization models mentioned above, a discrete event simulation based model that analyzes the impact of strategy changes in car sharing systems is introduced by Fassi et al. [27]. The strategies considered include opening new stations, increasing station capacities, merging or splitting stations. The model is evaluated on a car sharing data from Montreal, Canada.

Location of recharging stations in electric car sharing systems is first studied by Boyacı et al. [10]. The authors focus on a multi-objective problem that combines strategic and tactical decisions in a one-way electric car sharing system. One of the objectives is to maximize the net profit of the operating company and the other one is to maximize the net benefit of the clients. The authors propose a mixed integer linear programming formulation for solving this problem but due to the intractability of this formulation caused by the large number of relocation variables, they employ an aggregated demand structure, where the trips (demand) with the same origin and destination centers are grouped together to decrease the number of decision variables. Demand centers are the clusters of origin/destination points of trips that can be accessed

by the same set of stations. Moreover, after each usage, the vehicles are charged for a fixed period of time and depending on the demand served by vehicles, these charging periods are calculated to reach the required battery level e.g. full battery. The proposed method is tested on a real data from Nice, France.

In a recent paper, Çalık and Fortz [16] focus on location of stations in a one-way electric car sharing where the demand is known in advance. The problem includes decisions on location and number of the stations, initial level of cars at operated stations, the subset of requests to serve, and their service routes. The authors provide a mathematical formulation that maximizes the profit of the operator by taking into account the revenue obtained from the served demand and the fixed costs of stations and car purchases. They conduct experiments on large scale problem instances obtained through a real data of Manhattan (New York, USA) taxi trips. In this paper, we extend the mathematical formulation of Çalık and Fortz [16] to the case where there is uncertainty in demand and based on this formulation, we develop a decomposition algorithm that can solve large scale problems in reasonable amount of time. Another recent paper by Biesinger et al. [7] provides a bi-level heuristic algorithm that aims to find the location of recharging stations and the initial number of cars in an electric car-sharing system. The algorithm is tested on real-world instances from Vienna, Austria.

We are also aware of a few recent works [13, 14] that provide heuristics and formulations for location of stations in electric car sharing systems. Among them, Brandstätter et al. [13] consider a system with perfect demand information. The authors provide integer programming formulations and construction heuristics which maximize the profit under budget constraints. They conduct experiments on randomly generated grid-graph instances and instances from Vienna (relatively larger in size). Brandstätter et al. [14] extend the methods of Brandstätter et al. [13] to the case with demand uncertainty where demand is represented by multiple scenarios. They consider a maximum walking time of five minutes and seven scenarios in their experiments on the Vienna instances. The model is tested on small-size grid-graph instances (maximum 100 requests, 50 stations, 5 scenarios) and a reduced version of the Vienna instances (1060 requests, 201 stations, 7 scenarios). With this model, most instances are solved to optimality within the one-week time limit enforced by the authors.

Other than the car sharing systems, location of charging stations were also considered for private electric vehicles and urban taxi providers. The literature is relatively denser for these type of problems and it is possible to find both exact solution methodologies [32, 48, 21, 3, 20, 1, 2, 40] and heuristic approaches [34, 36, 47] for location of public charging units.

3. Problem definition and formulation

In this section, we present a formal definition and a mathematical formulation of our problem. Given a city network $G = (V, A)$ with arc set A, node set V, and set of potential stations $J \subset V$, we are required to select a subset $\overline{J}$ of J and locate an initial number of cars to the selected stations in order to satisfy a part of the customer demand represented by multiple scenarios consisting of individual requests. Each request must be served by a car available at a station accessible from the origin node at the time of departure and the car should be left at an arrival station accessible from the destination of the request with a free parking/charging spot at the time of arrival. Additionally, the battery consumption between the departure and the arrival stations assigned to a request should not exceed the battery level of the car and the total length (time) of the trip from the origin node to the destination should not be larger than a given threshold

Δ_k for request k. The traveling time and the energy consumption for each request are computed based on the shortest path between the departure and arrival stations allocated. One can also consider the case where these values are known explicitly for each request. A car that completes its service is plugged into a charger at its arrival station and it does not become available until the beginning of the first period after it has full battery level. For any scenario, the total number of cars (available or being charged) at a station should not exceed the capacity of that station at any time. The objective of the problem is to maximize the total expected profit that is equivalent to the total expected revenue obtained from the requests served minus the total cost of stations selected and cars owned.

Before going into the details of our mathematical formulation, we provide an initial set of parameter definitions used throughout the paper here. Upper case letters denote sets whereas lower case ones denote indices e.g. J is a set and j is the index of an element in J.

- $V = \{1, \ldots, n\}$ is the set of nodes.
- $J = \{1, \ldots, m\}$ is the set of potential stations where $J \subset V$.
- f_j is the fixed cost of locating a station on node $j \in J$.
- g is the cost of purchasing a car.
- C_j is the capacity of station $j \in J$.
- $T = \{0, \ldots, \tau\}$ is the set of time slots (identical length)
- $S = \{1, 2, \ldots, |S|\}$ is an index set corresponding to the scenarios.
- q_s is the probability that scenario $s \in S$ will occur.
- K_s is the set of requests in scenario $s \in S$, with origin $O_k \in V$, destination $D_k \in V$, starting time $T_k \in T$, and revenue p_k for $k \in K_s$. $K = \bigcup_{s \in S} K_s$.
- δ_{ij} is the battery usage (Ws) on the way from station $i \in J$ to station $j \in J$.
- β is the battery capacity (Ws) for each car .
- d_{ij} is the travel time from station $i \in J$ to station $j \in J$.
- d^w_{ij} is the walking time from node $i \in V$ to node $j \in S$.
- β^w is the maximum walking time between the origin (destination) points of requests and the departure (arrival) stations they are assigned to.
- Δ_k is the maximum time that a trip assigned to customer k can take (including the walking time).

We formulate our problem as a two stage stochastic program, in which location of stations and initial number of cars at each station are the first stage decisions whereas the requests served within each scenario and allocation of stations to the requests are second stage decisions. In this paper, we focus on the deterministic equivalent of our model. We propose a path based formulation with a set of decision variables associated with the paths (ordered list of nodes to visit) allocated to the requests.

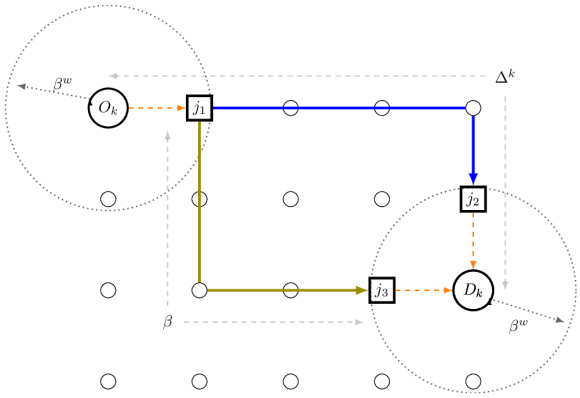


Figure 1: Feasible trips generated for a single request

We initially apply a preprocessing procedure to generate a set H_k of all feasible paths (trips) for each request $k \in K$. A path is feasible if it satisfies the accessibility, total length, and battery restrictions. Define $\bar{H}_s = \bigcup_{k \in K_s} H_k$ for scenario $s \in S$ and $H = \bigcup_{k \in K} H_k = \bigcup_{s \in S} \bar{H}_s$, then for each $h \in H$, we know the stations to visit on the trip (P_h) together with the time of visit, the amount of time required to recharge the car at the arrival station, thus, the time slot that the car will be ready for another customer. We illustrate this procedure for a single request on a simple example in Figure 1. In this figure, we use a grid network and rectilinear distances. For a request k with origin O_k and destination D_k, we illustrate two feasible trips. The nodes of the grid network that fall within each dashed circle of radius β^w are the accessible points from/to O_k or D_k. Squares represent potential stations and in this case, j_1 is the only potential station accessible to O_k whereas j_2 and j_3 are both accessible to D_k. As we consider only a single path between two stations, we can have two feasible trips for request k: Trip 1 (Path 1) is $O_k \to j_1 \to j_2 \to D_k$ if the total time spent is no greater than Δ_k and $\delta_{j_1 j_2} \leq \beta$; and Trip 2 (Path 2) is $O_k \to j_1 \to j_3 \to D_k$, similarly, if the total time spent is no greater than Δ_k and $\delta_{j_1 j_3} \leq \beta$. We generate all such feasible trips for each request during the preprocessing.

Our methods can be easily adapted to more challenging variants of our problem with slight modifications on this preprocessing procedure. One of these variants allows customers to visit intermediate stations when it is not possible to serve a request without recharging the battery or changing the car due to the long distance between the origin and destination nodes. We present a pseudo-code of our pre-processing procedure for this generalized case in [16]. We do not consider visits to intermediate stations in the computational results of this paper as we focus on the application inside cities where it is unlikely to have these type of requests. Another possible extension would be to consider time dependent traveling times and energy consumptions in the problem.

After the pre-processing, we retrieve the following parameters:

- $b^t_{hj} = 1$ if the car used for trip $h \in \bar{H}_s$ of scenario $s \in S$ exits station $j \in J$ at time $t \in T$, 0 otherwise.
- $\mu^t_{hj} = 1$ if the car used for trip $h \in \bar{H}_s$ of scenario $s \in S$ is being recharged at station $j \in J$ at time $t \in T$, 0 otherwise.

- $\lambda_{hj}^t = 1$ if the charging of the car used for trip $h \in \bar{H}_s$ of scenario $s \in S$ is completed at station $j \in J$ at time $t \in T$, 0 otherwise.

Then, we define the following decision variables for our formulation:

- $u_h = 1$ if trip $h \in \bar{H}_s$ of scenario $s \in S$ is chosen, 0 otherwise.
- L_j^{ts} is the number of available cars at station $j \in J$ at the beginning of time $t \in T$ for scenario $s \in S$.
- L_j^0 is the number of available cars at station $j \in J$ at time 0.
- $y_j = 1$ if a station is located at node $j \in J$, 0 otherwise.

Now, we can write our path based formulation (PF) as follows:

$$(PF) \qquad \max \sum_{s\in S} q_s \sum_{h\in \bar{H}_s} p_h u_h - \sum_{j\in J} f_j y_j - g \sum_{j\in J} L_j^0 \tag{1}$$

$$\text{s.t.} \sum_{h\in H_k} u_h \leq 1, \qquad \forall k \in K \tag{2}$$

$$u_h \leq y_j, \qquad \forall j \in J, h \in H : j \in P_h \tag{3}$$

$$\sum_{h\in \bar{H}_s} b_{hj}^t u_h \leq L_j^{ts}, \qquad \forall j \in J, t \in T, s \in S \tag{4}$$

$$L_j^{ts} + \sum_{h\in \bar{H}_s} (\mu_{hj}^t - b_{hj}^t) u_h \leq C_j y_j, \qquad \forall j \in J, t \in T, s \in S \tag{5}$$

$$L_j^{ts} = L_j^{(t-1)s} + \sum_{h\in \bar{H}_s} (\lambda_{hj}^t - b_{hj}^{(t-1)}) u_h, \qquad \forall j \in J, t \in T : t \geq 1, s \in S \tag{6}$$

$$L_j^{0s} = L_j^0, \qquad \forall j \in J, s \in S \tag{7}$$

$$0 \leq L_j^{ts} \leq C_j y_j, \qquad \forall j \in J, t \in T, s \in S \tag{8}$$

$$L_j^0 \in \mathbb{Z}_0^+, \qquad \forall j \in J \tag{9}$$

$$u_h \in \{0,1\}, \qquad \forall h \in H \tag{10}$$

$$y_j \in \{0,1\}, \qquad \forall j \in J \tag{11}$$

The objective function (1) maximizes the expected profit. The first term in this function gives the expected revenue obtained by serving the customers, the second term is the total fixed cost of opened stations, and the third term is total cost of purchasing cars.

By Constraints (2), a customer is served with at most one trip and by Constraints (3) every station on a selected trip is forced to be opened.

For each scenario, Constraints (4)-(8) ensure the following restrictions: Constraints (4) restrict the number of cars leaving a station with the number of available cars at that station for each time slot. Constraints (5) ensure that the capacity of each station is respected, so that parking a car is not allowed if there is no free space at the station. Constraints (6) balance the number of cars at each station at each time slot. Constraints (7) initialize the number of cars available for each station and Constraints (8) restrict this number with the capacity of that station for each time zone.

Finally, Constraints (9)-(11) give the non-negative integrality and binary restrictions.

In the following section, we present the test results on our formulation PF, its LP relaxation, and the partial relaxation RPF where we keep y variables as binary and relax the integrality of u variables as follows:

$$
\begin{aligned}
(RPF) \qquad & \max \ (1) \\
& \text{s.t. } (2)-(9),(11) \\
& \quad 0 \leq u_h \leq 1, \qquad\qquad \forall h \in H \qquad\qquad (12)
\end{aligned}
$$

4. Computational study on PF and its relaxations

In order to test our methods, we generate instances with multiple scenarios based on a real data obtained from Manhattan (New York, USA) taxi trips [49, 44]. The data file is based on a city network in Manhattan with 10556 nodes, 85 potential station locations, and 25592 arcs. It contains 27549 requests with origin, destination, starting time, revenue, and duration; the capacity of each station (maximum number of recharging units); the base cost of opening each station, per unit cost of installing a slow or fast charging unit, and purchase cost of three types of electric cars; charging speed for both types of charging units; and the distance and time dependent maximum traveling speed for each arc. Based on the information given in the Manhattan data file, we are able to obtain the values of all the parameters of an instance.

In our experiments, we consider a single car type (Smart ED), a period of 5 minutes, a capacity of a station identical to its number of charging units, and only fast charging units. Through our demand forecasting method (detailed in [17]), we generate instances that include requests with different origin-destination pairs than the ones in the original data file. We compute the revenue of these newly generated requests by using the formula of the original data (0.3 € per minute) based on the shortest path distances between the origin and the destination nodes. By using a fixed driving speed of 50 km/h and a walking speed of 1.34 m/s, we obtain the shortest paths for walking and traveling times via Dijkstra's algorithm. Each one of the scenarios includes requests coming from a one-day period. In order to see how the results change with different scenario combinations, we create four different sets of scenarios (C1-C4) generated from the same probability distribution.

The cost of a Smart ED in the data used in our experiments is 20000 €. Its battery capacity is 63360 kWs and fast charging rate is 17600 W. The base cost of stations ranges between 9000 € and 64000 € . The cost of fast charging slots ranges from 22000 € to 32000 € per unit.

We conduct our experiments by using IBM ILOG CPLEX 12.7 in the Java environment on an Intel(R) Xeon(R) E5-2630 v3 CPU at 2.40GHz with 16 cores and 32 GB of RAM.

For the Dijkstra's algorithm, we use the implementation of the JGraphT 1.0.1 package. We impose a one hour time limit and 16 GB memory limit to each experiment. We keep CPLEX cuts and presolve on as they seemed to be efficient in our experiments.

In our preliminary experiments, we observe that the cost values in the original data are too high to have a profitable system. A similar issue is observed also by [14] in their case study for ECS systems in Vienna. This indicates the need for a comprehensive study on pricing and additional income resources for the initial investments of ECS systems but this remains out of the scope of our paper. In order to have solutions that open stations and serves customers, we introduce a parameter *cost factor* that divides the cost values of the original data. This way we try to preserve the distinct cost values for stations at the same proportion. This parameter can also be considered as expected repetitions of similar requests in the long run taking

into account the lifetime of stations and cars. In our experiments, we use two different values, namely, 10^5 and 10^6 as *cost factor*. We also conducted experiments with *cost factor* of 10^4, which is the realistic value obtained by considering the depreciation, but the system did not provide positive profits for such high costs. Similar results were observed also in our previous study for pre-booked ECS systems [16]. We also tested our instances with distinct *cost factor* values for cars and stations (10^5 for stations and less than 10^5 for cars), but we ended up with non-profitable systems. Therefore, we do not report them here. Moreover, we use $\Delta_k = 1.1 \times d_{O_k D_k}$ for every $k \in K$. Δ_k is the time including walking times and driving times between the stations assigned. The walking times are already restricted by β^w parameters and customers cannot be forced/expected to walk for longer than β^w. This type of restriction allows longer driving time (rather than longer walking time) if the driving time between O_k and D_k is longer.

In this section, we show results regarding PF, its LP relaxation, and RPF for $costfactor$ values 10^5 (High cost) and 10^6 (Low cost) in Figures 3 - A.15. We provide tables with detailed results for one copy of scenarios, namely for C1, in the Appendix A.

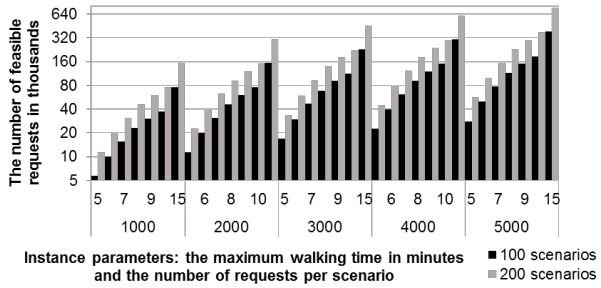
In these figures and tables, $|K_s|$ denotes the number of requests in each scenario generated and this number is the same for every scenario of a single problem instance. The β^w column gives the maximum walking time restriction in minutes, all other time values are given in seconds. The columns labeled 'Obj' give the solution value obtained from the corresponding method at termination due to optimality or resource limitations (time or memory). We report two types of gap percentages: $g1(\%)$ gives the gap relative to the best known PF solution value, say Obj^*, then, $g1(\%) = 100 \times [(Obj - Obj^*)/Obj^*]$; and $g2(\%)$ denotes the gap provided by CPLEX at the termination. We here note that for some instances, Obj^* value might be obtained via the methods that we propose in Section 5. We further indicate the instances that cannot be solved to optimality within the time limit with 'TL' under the 'time' column. We do not hit the memory limit in any of the instances. If we are not able to obtain a feasible solution or the gaps within the time limit for an instance, we indicate this as 'NA'. As the instances with $\beta^w = 15, |K_s| \geq 3000$ are intractable, we omit corresponding problems for $|S| = 200$ charts and tables.

In Figure 2, we present the results related to pre-processing, the initial process to generate all feasible trips, on problems with $|S| = 100$ and $|S| = 200$ for the first set of scenarios (similar numbers for the others). On the X axis of these charts, we have instances corresponding to $\beta^w, |K_s|$ combinations. On the Y-axis, we use a logarithmic scale to show $|K_a|$, the total number of requests that satisfy accessibility and maximum length restrictions over all scenarios (Figure 2a); and $|H|$, the total number of trips generated i.e. the number of $\mathbf{u}$ variables in the model (Figure 2b). Here, we observe that even one minute of increase on the accessibility measure (β^w) can double the number of requests that can be potentially served. The total time spent during the pre-processing of these instances ranges from 30 seconds to half an hour.

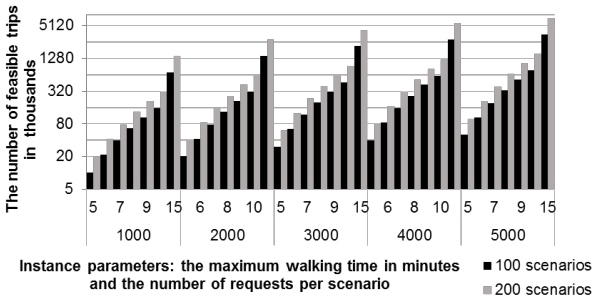
In Figure 3, we observe that the LP relaxations provide very tight bounds, especially, for instances of low costs. The largest LP gap that we observe is 6.98% and it belongs to the C4 instance with $\beta^w = 5, |K_s| = 1000, |S| = 100$ and high costs. The average LP gap of all instances tested is 0.39%. Although there are a few exceptions, usually the LP gaps tend to decrease as β^w increase.

When we look at Figure 4 of instances with $|S| = 100$, we can observe a pattern of increase in the objective values in parallel to the increase in β^w as well as in $|K_s|$. Moreover, we see that these objective values are higher when we have low cost parameters. We observe a similar pattern for C2-C4 instances.

We present the solving times of LP, RPF, and PF in Figures A.13-A.15 of the Appendix. As expected,



(a) The total number of requests that satisfy accessibility and maximum length restrictions.



(b) The number of feasible trips generated.

Figure 2: Preprocessing results

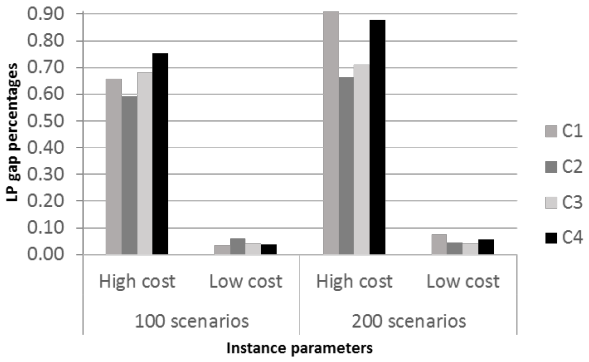


Figure 3: Average LP gaps (g_1%) for C1-C4

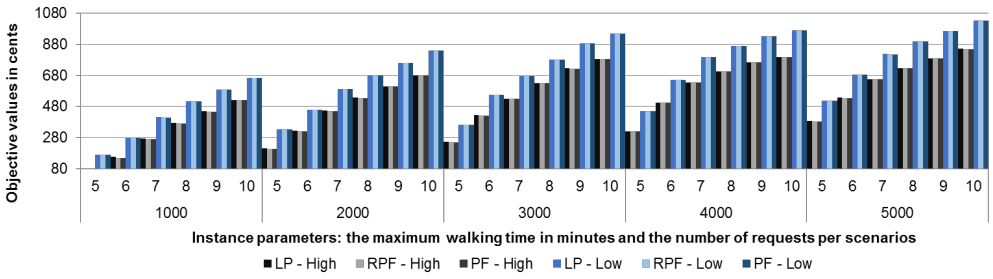


Figure 4: LP, RPF, and PF values of C1 for 100 scenarios. Blue bars for low cost, gray bars for high cost.

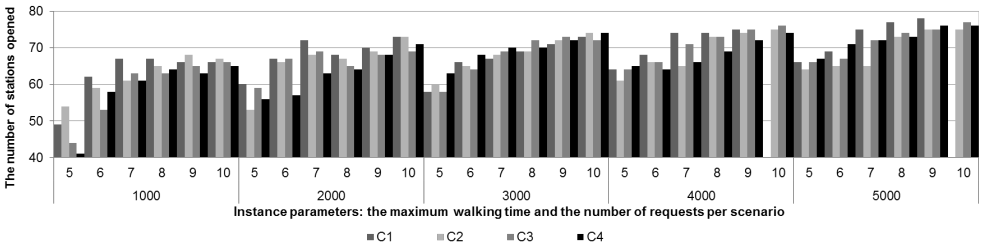


Figure 5: Number of stations opened per instance for C1-C4 for 100 scenarios and high cost.

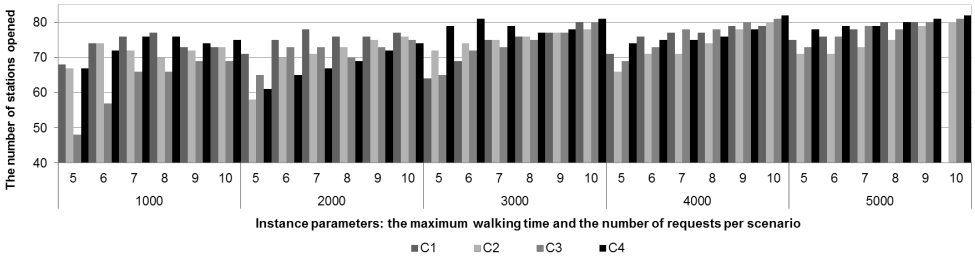


Figure 6: Number of stations opened per instance for C1-C4 for 100 scenarios and low cost.

PF spends a larger amount of time to solve problems with larger β^w values. PF is not able to find any solution with a positive profit value in instances with $\beta^w = 15$ mins and it spends all the time given for the presolve of CPLEX when $|K_s| \geq 2000$ for $\beta^w = 15$. For some of the instances, RPF needs more time to reach optimality than PF.

Moreover, we present the number of stations opened for the solution obtained from PF in Figures 5 and 6 for $|S| = 100$. An important observation here is that the increase in the number of stations opened when β^w is increased from 5 minutes to 6 minutes is more significant compared to other increments, for example, from 6 minutes to 7 minutes. This is worth to express as most recent works in the literature restrict the walking time by 5 minutes but we see through our experiments that a one-minute increase of this parameter might cover a much larger portion of the demand and yield a significant increase in the expected profit.

Another important observation is on the number of trips that can be served by the solution of PF. We show these values for $|S| = 100$ in Figures 7 and 8. Here, we can see that the number of trips that can be served increases as β^w increases, and usually also when $|K_s|$ increases.

Our experiments reveal that the computational difficulty increases parallel to the number of scenarios

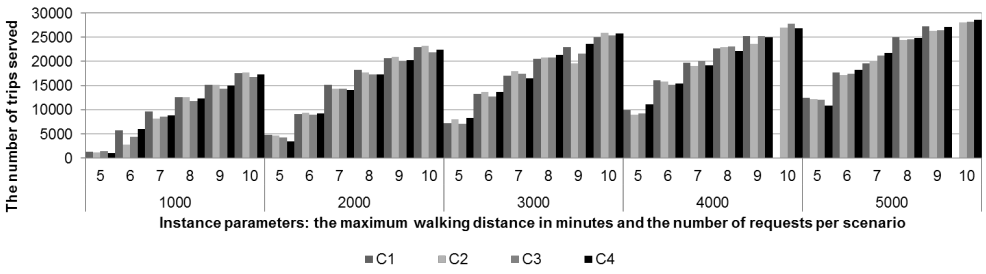


Figure 7: The number of trips that can be served per instance for C1-C4. $|S| = 100$, high cost.

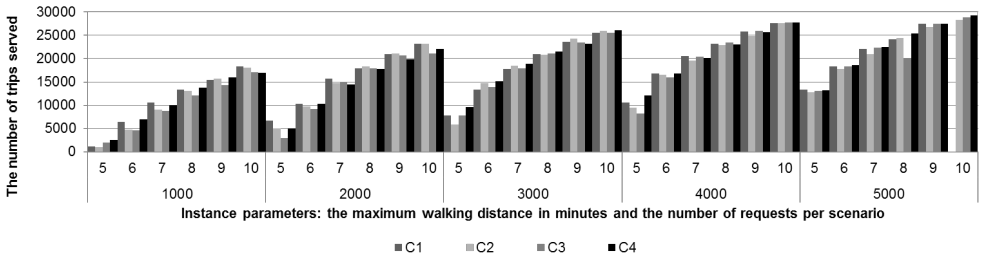


Figure 8: The number of trips that can be served per instance for C1-C4. $|S| = 100$, low cost.

as the number of constraints and variables increases significantly. In order to overcome this difficulty, we describe a Benders decomposition algorithm in the following section.

Another interesting observation on these results is that the objective value of RPF is the same as the objective value of PF for almost every instance and the difference is very small for the nonidentical ones. As it will be explained in the next section, this observation encourages us to focus on two phase solution methodologies that will first focus on solving RPF which has less number variables with integrality restrictions compared to PF and then reach to the optimality of PF by using the information obtained from RPF.

5. Decomposition algorithm (BAC)

Before presenting the details of our algorithm, we give a general description of the Benders decomposition method [6]. For a mixed integer program with a group of integer variables and a group of continuous variables, the algorithm consists of solving the formulation by temporarily removing all continuous variables. This problem is called the master problem. Then, the Benders algorithm focuses on the dual of a subproblem, which is simply a restriction of the original problem where all the integer variables are fixed to the values of the master problem. From the dual problem, a so called feasibility cut for each extreme ray and an optimality cut for each extreme point is obtained. These cuts are added to the master problem to obtain another solution and the procedure is repeated. Since enumeration of all extreme rays and extreme points is not very practical, cutting plane algorithms are commonly used in the literature.

The convergence rate of the classical Benders decomposition method might be very slow especially if the subproblem is difficult to solve. On the other hand, it might be very efficient if the subproblem can be decomposed further into smaller and easy-to-solve problems as in multi-commodity, multi-period, or multi-scenario problems [8]. Motivated by this fact, we aim further decomposing the second-stage problem of our two-stage recourse problem into smaller problems, each one corresponding to a single demand scenario. For this purpose, we decide to choose $\mathbf{y}, \mathbf{L^0}$ as first stage variables that are kept in the master problem and $\mathbf{u}, \mathbf{L}$ as the variables of the subproblem. However, as $\mathbf{u}$ variables are integral, we cannot apply the classical Benders decomposition to PF without any further adaptations. On the other hand, with this type of categorization of variables, RPF is a good candidate for applying the Benders procedure as $\mathbf{u}, \mathbf{L}$ variables are continuous in this model. In this case, we need additional steps to make sure that the optimal solution of our algorithm provides an integral vector $\mathbf{u}$. Several successful adaptations of the classical Benders decomposition algorithm to different IP problems are available due to [9, 30, 37]. In our implementation, we use the "Combinatorial Benders cuts", which are first named by Codato and Fischetti [23].

We implemented our Benders algorithm as a two phase method. In the first phase of our algorithm, we solve RPF with a classical Benders framework. Let $v(RPF)$ be the optimal value of RPF and $(\mathbf{y}^*, \mathbf{L}^*)$ be the optimal values of the first stage variables $(\mathbf{y}, \mathbf{L^0})$. Then, we solve PF by fixing values of $\mathbf{y}, \mathbf{L^0}$ variables to $(\mathbf{y}^*, \mathbf{L}^*)$ to see if there exists an integer feasible solution for RP with objective value $v(RPF)$. If yes, we stop as we have an optimal solution of PF on hand. Otherwise, we move to the second phase of the algorithm. The motivation behind this first phase is the fact that RPF gives the optimal value of PF in most instances; so, the values of the first stage decision variables obtained from RPF might possibly provide a complete integral solution for PF and avoid excessive number of integrality checks for addition of relatively weak combinatorial cuts.

In Phase II, we restart a branch-and-cut framework for RPF and restrict the objective value from above with $v(DRPF)$, the value of the best dual bound obtained in Phase I. Then, for integer solutions of RPF tree, we solve individual subproblems with integrality restrictions on $\mathbf{u}$ variables for each scenario and add combinatorial Benders cuts if these subproblems are integer infeasible. Our experimental study reveals that the two-phase procedure converges faster than introducing the combinatorial cuts at the first phase in most instances and it is much more efficient in overall as most problems do not require the second phase.

Now we present the details of Phase I and Phase II of our algorithm.

Phase I:

Through our experiments, we observed that the optimality cuts worsen the solving time of the algorithm so, we decided to avoid them. To do so, we make a modification on our formulation so that we obtain only feasibility cuts from our dual subproblem. We define a nonnegative decision variable z_s for each scenario $s \in S$ and ensure that its value will be equal to the revenue obtained from that scenario as follows:

$$(PF2) \qquad \max \sum_{s\in S} q_s z_s - \sum_{j\in J} f_j y_j - g\sum_{j\in J} L_j^0 \tag{13}$$

$$\text{s.t.} \sum_{h\in \bar{H}_s} p_h u_h \geq z_s, \qquad \forall s \in S \tag{14}$$

$$(2) - (11)$$

$$0 \leq z_s \leq \sum_{k\in K_s} p_k, \qquad \forall s \in S \tag{15}$$

To solve $(PF2)$ in a Benders fashion, we keep $\mathbf{z}, \mathbf{y}, \mathbf{L}^0$ variables in the master problem and deal only with $\mathbf{u}$ variables in the subproblem.

In order to make our master problem stronger, we introduce two sets of valid inequalities (18) and (19). Thus, we solve (MP) given below as the master problem of our Benders algorithm.

$$(MP) \qquad \max \ \sum_{s\in S} q_s z_s - \sum_{j\in J} f_j y_j - g\sum_{j\in J} L_j^0 \tag{16}$$

$$\text{s.t.} \ \sum_{m=1}^{C_j} x_{jm} \le y_j, \qquad \forall j \in J \tag{17}$$

$$z_s \le \sum_{k\in K_s} p_k \sum_{j\in N_{O_k}} y_j, \qquad \forall s \in S \tag{18}$$

$$z_s \le \sum_{k\in K_s} p_k \sum_{j\in N_{D_k}} y_j, \qquad \forall s \in S \tag{19}$$

$$z_s \ge 0, \qquad \forall s \in S \tag{20}$$

$$(9), (11)$$

In this formulation, N_{O_k} and N_{D_k} denote the set of stations accessible by, respectively, origin and destination, of request $k \in K$. Once we introduce fixed values $(\overline{\mathbf{z}}, \overline{\mathbf{y}}, \overline{\mathbf{L}}^0)$ to our subproblem, we observe that it can be decomposed into $|S|$ problems, one for each scenario, that can be solved independently.

At this point, we make some simplifications and modifications to have smaller number of constraints and variables in our primal and dual subproblems. The first observation is as follows:

Property 1. *Constraints (8) are redundant.*

Proof. (i) $L_j^{ts} \ge 0, \forall j \in J, t \in T, s \in S$:
By Constraints (4), $L_j^{ts} - \sum_{h\in \bar{H}_s} b_{hj}^t u_h \ge 0, \forall j \in J, t \in T, s \in S$. Then, by Constraints (6), $L_j^{ts} = L_j^{(t-1)s} - \sum_{h\in \bar{H}_s} b_{hj}^{(t-1)} u_h + \sum_{h\in \bar{H}_s} \lambda_{hj}^t u_h \ge 0, \forall j \in J, t \in T, s \in S$ as $\lambda_{hj}^t, u_h \in \{0,1\}, \forall j \in J, t \in T, h \in H$.

(ii) $L_j^{ts} \le C_j y_j, \forall j \in J, t \in T, s \in S$:
By Constraints (5), $\sum_{h\in \bar{H}_s} \mu_{hj}^t u_h \le C_j y_j + \sum_{h\in \bar{H}_s} b_{hj}^t u_h - L_j^{ts}, \forall j \in J, t \in T, s \in S$. Moreover, $\sum_{h\in \bar{H}_s} \lambda_{hj}^{(t+1)} u_h \le \sum_{h\in \bar{H}_s} \mu_{hj}^t u_h, \forall j \in J, t < \tau$ as the number of cars who finished recharging at $t+1$ cannot be larger than the number of cars being charged at t. Then, $\sum_{h\in \bar{H}_s} \lambda_{hj}^{(t+1)} u_h + L_j^{ts} - \sum_{h\in \bar{H}_s} b_{hj}^t u_h \le C_j y_j$ implying that $L_j^{(t+1)s} \le C_j y_j, \forall j \in J, t \in T : t < \tau, s \in S$ via Constraints (6).

□

Moreover, the following replacement of L_j^{ts} is possible due to Constraints (6) and (7):

$$L_j^{ts} = L_j^0 + \sum_{h\in \bar{H}_s} (\sum_{r=1}^{t} \lambda_{hj}^t - \sum_{r=0}^{t-1} b_{hj}^t) u_h, \qquad \forall j \in J, t \in T : t \ge 1, s \in S \tag{21}$$

If we replace L_j^{ts} with the right hand side of (21) throughout PF2 and omit equality Constraints (6) and (7), we obtain a formulation with $\mathbf{z}, \mathbf{y}, \mathbf{u}, \mathbf{L^0}$ variables only. Then, for each scenario $s \in S$, we can express the corresponding subproblem (SP_s) as follows:

$$(SP_s) \qquad \max\ 0 \tag{22}$$

$$\text{s.t.} \sum_{h\in\bar{H}_s} p_h u_h \geq \overline{z}_s, \tag{23}$$

$$\sum_{h\in H_k} u_h \leq 1, \qquad \forall k \in K_s \tag{24}$$

$$u_h \leq \overline{y}_j, \qquad \forall h \in \bar{H}_s, j \in P_h \tag{25}$$

$$\sum_{h\in\bar{H}_s} (\sum_{r=0}^{t} b_{hj}^r - \sum_{r=1}^{t} \lambda_{hj}^r) u_h \leq \overline{L}_j^0, \qquad \forall j \in J, t \in T \tag{26}$$

$$\sum_{h\in\bar{H}_s} (\sum_{r=1}^{t} \lambda_{hj}^r - \sum_{r=0}^{t} b_{hj}^r + \mu_{hj}^t) u_h \leq C_j \overline{y}_j - \overline{L}_j^0, \qquad \forall j \in J, t \in T \tag{27}$$

$$u_h \in \{0,1\}, \qquad \forall h \in \bar{H}_s \tag{28}$$

Let LP_s denote the linear programming (LP) relaxation of SP_s, where the integrality restriction on $\mathbf{u}$ variables is removed, therefore, $0 \leq u_h \leq 1, \forall h \in \bar{H}_s$; DLP_s denote the dual of LP_s; and $\alpha, \epsilon_k, \gamma_{jh}, \theta_{jt}$, and ω_{jt} be the dual variables associated with Constraints (23)-(27), respectively. Here, we note that $u_h \leq 1$ restriction is already satisfied due to (24) and (25) $\forall h \in \bar{H}_s$, so, we do not include an additional set of constraints for this restriction. Then, we can express DLP_s as follows:

$$(DLP_s) \qquad \min\ -\overline{z}_s\alpha + \sum_{k\in K_s} \epsilon_k + \sum_{h\in\bar{H}_s, j\in P_h} \overline{y}_j \gamma_{hj} + \sum_{j,t} \overline{L}_j^0 \theta_{jt} + \sum_{j,t} (C_j \overline{y}_j - \overline{L}_j^0)\omega_{jt} \tag{29}$$

$$\text{s.t.} -p_h\alpha + \sum_{k:h\in H_k} \epsilon_k + \sum_{j\in P_h} \gamma_{hj} + \sum_{j,t} (\sum_{r=0}^{t} b_{hj}^r - \sum_{r=1}^{t} \lambda_{hj}^r)\theta_{jt} + \sum_{j,t} (\sum_{r=1}^{t} \lambda_{hj}^r - \sum_{r=0}^{t} b_{hj}^r) + \mu_{hj}^t)\omega_{jt} \geq 0, \qquad \forall h \in \bar{H}_s \tag{30}$$

$$\alpha \geq 0, \tag{31}$$

$$\epsilon_k \geq 0, \qquad \forall k \in K_s \tag{32}$$

$$\gamma_{hj} \geq 0, \qquad \forall h \in \bar{H}_s, j \in P_h \tag{33}$$

$$\theta_{jt}, \omega_{jt} \geq 0, \qquad \forall j \in J, t \in T \tag{34}$$

In our implementation, we introduce $\alpha \leq 1$ to (DLP_s) and associate unboundedness of the problem with a negative objective value. Therefore, as soon as the optimal value obtained from (DLP_s) is negative, we add feasibility cuts (35) to our master problem. An important property of DLP_s is that the feasible region of the model does not change by the solution of the master problem. We use this property by generating the individual dual problems once in our implementation and changing only the objective function at each

iteration. This modification in the implementation improves the efficiency of our algorithm significantly.

$$-\overline{\alpha}z_s + \sum_j (\sum_{h:j\in P_h} \overline{\gamma}_{hj} + C_j \sum_t \overline{\omega}_{jt}) y_j + \sum_j (\sum_t \overline{\theta}_{jt} - \sum_t \overline{\omega}_{jt}) L_j^0 \geq -\sum_{k\in \bar{H}_s} \overline{\epsilon}_k \tag{35}$$

In addition to every integral solution of the branch-and-bound tree, we add feasibility cuts (35) for violated fractional solutions at the root node as well. We use the *LazyConstraintCallback* and *UserCutCallback* of CPLEX for separation of, respectively, integral and fractional solutions.

After this branch-and-cut first phase, the feasibility of the optimal solution is checked as follows. Let $v(RPF)$ be the optimal value of RPF and $(\mathbf{y}^*, \mathbf{L}^*)$ be the optimal values of the first stage variables $(\mathbf{y}, \mathbf{L^0})$. Then, we solve PF by fixing values of $(\mathbf{y}, \mathbf{L^0})$ variables to $(\mathbf{y}^*, \mathbf{L}^*)$ to see if there exists an integer feasible solution for RP with objective value $v(RPF)$. If yes, we stop as we have an optimal solution of PF on hand. Otherwise, we move to the second phase of the algorithm.

Phase II:

In order to start Phase II, we make use of $v(DRPF)$ and $(\mathbf{y}^*, \mathbf{L}^*)$ obtained from Phase I. At this stage, we introduce a new set of binary variables $\mathbf{x}$ that will represent integral $\mathbf{L^0}$ variables with the variable transformation $L_j^0 = \sum_{m=1}^{C_j} m x_{jm}, j \in J$ in MP. Doing so, we can introduce combinatorial Benders cuts associated with these variables when needed. We further add the following constraints to MP:

$$\sum_{s\in S} q_s z_s - \sum_{j\in J} f_j y_j - g \sum_{j\in J} \sum_{m=1}^{C_j} m x_{jm} \leq v(DRPF) \tag{36}$$

$$\sum_{m=1}^{C_j} x_{jm} \leq y_j, \qquad \forall j \in J \tag{37}$$

Similar to Phase I, we solve DLP_s for separation not only at integer solutions of the branch-and-bound tree but also at fractional solutions at the root node. But this time, the separation cut would be (38).

$$-\overline{\alpha}z_s + \sum_j (\sum_{h:j\in P_h} \overline{\gamma}_{hj} + C_j \sum_t \overline{\omega}_{jt}) y_j + \sum_j (\sum_t \overline{\theta}_{jt} - \sum_t \overline{\omega}_{jt}) \sum_{m=1}^{C_j} m x_{jm} \geq -\sum_{k\in \bar{H}_s} \overline{\epsilon}_k \tag{38}$$

Additionally, once DLP_s is bounded (i.e. LP_s feasible) for every $s \in S$, we check if we have an integral solution for $SPMax_s$ for the current $(\overline{\mathbf{y}}, \overline{\mathbf{x}})$ values of the master problem iteratively for $s = 1, \ldots, |S|$.

$$(SP_sMax) \qquad \max \sum_{h\in \bar{H}_s} p_h u_h \qquad \text{s.t. } (24) - (28). \tag{39}$$

If $SPMax_s$ is infeasible for some $s \in S$, we add the combinatorial feasibility cut (40) and do not check integer feasibility for the remaining subproblems. If $SPMax_s$ is feasible but the optimal value is greater

than $\overline{z}_s$ for some $s \in S$, then we add the optimality cut (41). This cut requires either the value of z_s variable to be tightened by $\sum_{h \in \bar{H}_s} p_h u_h^*$ for the same $(\overline{\mathbf{y}}, \overline{\mathbf{x}})$ solution or to change the value of y_j for some $j \in J$ or x_{jm} for some (j, m) where $j \in J, m = 1, \ldots, C_j$. At the termination of this branch-and-cut procedure, we obtain an optimal solution for PF.

$$F(\overline{y}, \overline{x}) \geq 1 \tag{40}$$

$$z_s \leq \sum_{h \in \bar{H}_s} p_h u_h^* + \sum_{k \in K_s} p_k F(\overline{y}, \overline{x}) \tag{41}$$

where $\mathbf{u}^*$ is the solution vector obtained from $SPMax_s$ of $s \in S$ and

$$F(\overline{y}, \overline{x}) = \sum_{j:\overline{y}_j=0} y_j + \sum_{j:\overline{y}_j=1} (1 - y_j) + \sum_{j,m:\overline{x}_{jm}=0} x_{jm} + \sum_{j,m:\overline{x}_{mj}=1} (1 - x_{jm}).$$

We here note that we also implemented a classical Benders decomposition by leaving u variables in the master problem and tested it both with the automated Benders function of CPLEX and with our manual implementation and we observed that this type of decomposition had a poor performance.

6. Computational Study on BAC

In this section, we provide a detailed analysis of the experimental study on our algorithm. Before going into this analysis, we briefly describe our demand generation method as follows: In order to generate multiple demand scenarios which follow the pattern of the available data to certain extent, we fit the data into several probability distribution (mass) functions (PDF) from the literature. Among these PDFs, we choose the one that gives the minimum of least squared errors to generate scenarios, each with a certain number of requests. Finally, we compute the probability of scenarios based on the probability of individual requests in each scenario. The details of this method can be found in [17].

6.1. Results

In addition to the metrics reported in earlier charts and tables, we also report here the number of nodes explored in the branch-and-bound tree (nodes). Moreover, regarding our algorithms, we give the number of integral and fractional solutions separated during branch-and-cut procedures via columns 'lazy' and 'user', respectively. We tried turning off the presolve option of CPLEX in our instances but decided to keep it on as the solving time performance was worse without presolve.

When we compare the solving times of model PF and algorithm BAC for instances with $|S| = 100, 200$ (see Figures A.16-A.19 in Appendix A), we see that BAC performs better in tackling instances of C1 and C3 with large β^w values and instances with $|K_s| \geq 4000$. However, the problems with $\beta^w = 15$ become intractable also by BAC when $|K_s| \geq 3000$. When we test PF and BAC on instances with $|S| = 200$, we see that both PF and BAC needed more time on average to reach optimality. They hit the time limit for more instances but the average and maximum gaps from the best solution value were much larger for PF compared to BAC. The better performance of BAC compared to PF becomes more visible in these instances. Although the time spent by BAC for C2 and C4 seems larger compared to PF, BAC is able to find very high

quality solutions within one hour for most instances whereas PF struggles with finding a non-zero feasible solution in many of them (see Figure 10 for average gap percentages).

An interesting observation is that among the problems for which we obtain feasible solutions, less than 20% need a call to Phase II. The solving times that go up to 2 hours for BAC in Figures A.16-A.19 correspond to some of these instances for which Phase II was needed.

On average, BAC needs to visit more branch-and-bound nodes than PF and usually, it needs to separate a larger number of fractional solutions than integer solutions even though the fractional solutions are separated only at the root node.

We provide a detailed comparison of PF and BAC for C1 instances in Tables A.6-A.9 of Appendix A.

6.2. Stabilization

When we compare the performance of BAC on problems with $|S| = 200$ with those of $|S| = 100$, we observe that more iterations of separation are needed in Phase I of the algorithm and Phase II iterations are needed in a larger portion of problems. To overcome these difficulties, we introduce a stabilization procedure that enables addition of potentially stronger cuts for separation of infeasible solutions at individual iterations of our Benders algorithm. This method was originally proposed by Ben-Ameur and Neto [5] for cutting-plane and column generation algorithms and it was successfully adapted to several Benders decomposition implementations [29, 18, 42].

In the original form of our algorithm, for each optimal solution, say $\pi^m = (z^m, y^m, L^m)$, we solve the dual of subproblem $SP1$ for π^m and add a cut associated with π^m to our master problem if there is a violation and move to subproblem $SP2$ to solve its dual for π^m and repeat for each subproblem. Now, instead of moving to next subproblem immediately after finding a violation with the current one, we do the following: Let π^f be a global feasible solution to the original problem PF and $\varphi \in (0, 1)$. Differently the core point of [41], π^f is not necessarily an interior point and it can be any feasible point. We find another point $\pi^1 = \varphi\pi^m + (1 - \varphi)\pi^f$ that is located on the line segment between π^m and π^f, and check if the dual of the current subproblem is violated by this point π^1. If yes, then we obtain a potentially stronger cut that separates both π^m and π^1, so, we add the new cut to our master problem. Then, we move to another point $\pi^2 = \varphi\pi^1 + (1 - \varphi)\pi^f$ and check if there is a violation. If there is no violation, we move to another point $\pi^3 = \varphi\pi^1 + (1 - \varphi)\pi^2$ and we keep moving on the line segment between the last feasible point and the last separation point found for certain number of iterations. In our implementation, we move for only two iterations between π^m and π^f for each subproblem and then, we move to the next subproblem to repeat the same stabilization steps. We further choose $\varphi = 0.5$ in our experiments.

We illustrate the stabilization procedure on a sample polyhedra in Figure 9. Here, the convex hull of the master problem is shown with the dashed blue lines and inside that, we have the intersection, initially, with the first subproblem SP1. The solution of the master problem π^m is infeasible for SP1 and FC_1 is a feasibility cut that separates π^m. Now, we move to π^1 and find another feasibility cut FC_2 to separate π^1 as it is also infeasible for SP1. Then, we move to another point π^2, which turns out to be feasible for SP1. After two iterations, we add FC_1 and FC_2 to our master problem and move to the second subproblem SP2. As π^m is feasible for SP2, we do not get any feasibility cuts and we do not need any stabilization for SP2, so, we can move to the next subproblem if there is any; otherwise, to the next solution of the master problem.

A summary of the results comparing the performance of BAC with and without stabilization is presented in Figures 10 and 11. Even though there is not an absolute winner among the two methods, the stabilized

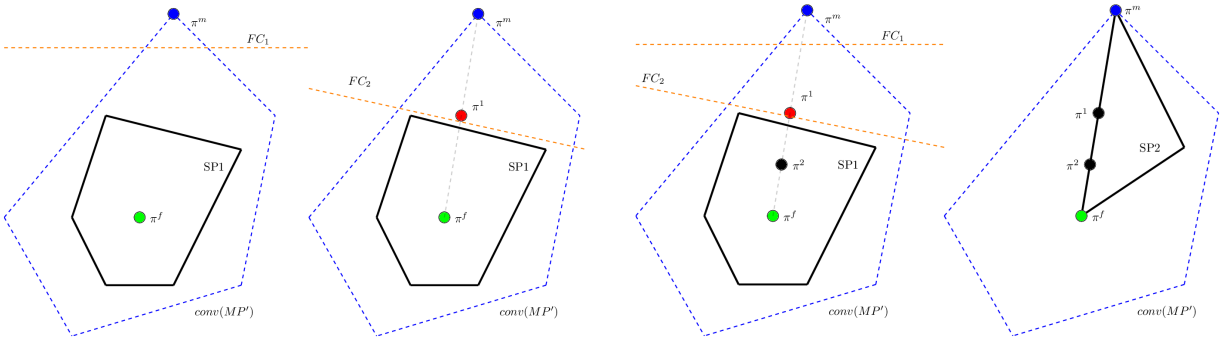


Figure 9: A sample stabilization iteration

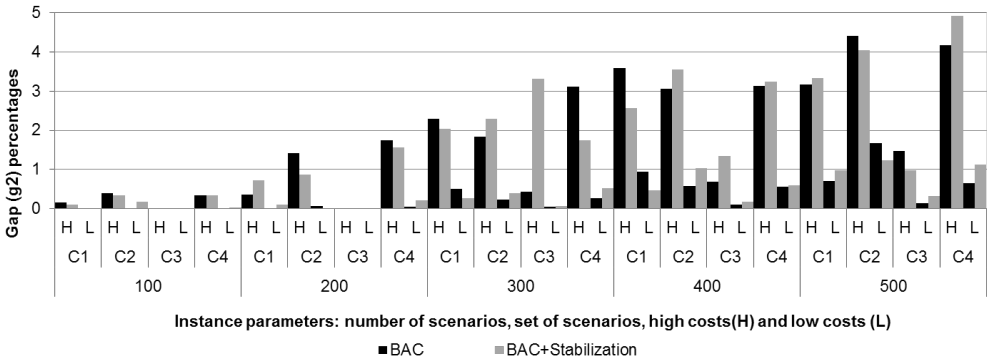


Figure 10: Average gaps ($g_2\%$) of our algorithms for C1-C4 instances

version performs slightly better on average solving time. It is also better on the average gap for C1 instances (1.18% vs 1.06%) whereas the non-stabilized version is better for C2-C4 instances and in overall average (1.06% vs 1.13%). Stabilization decreases the average number of *LazyConstraintCallback* iterations and it decreases the average number of *UserCutCallback* iterations by around 50% for each of C1-C4. The impact of stabilization on the number of nodes explored does not follow a clear pattern and it is difficult to draw any conclusion on this aspect (Detailed results regarding the stabilization experiments can be provided by the authors upon request).

6.3. Sensitivity Analysis

In this section, we provide a brief sensitivity analysis based on our observations regarding the effects of changes in parameters $costfactor$, β^w, $|S|$, $|K_s|$ on problem outputs.

When we look at the average gap % in Figure 10, we can easily see that it decreases when the cost values decrease and it usually increases when the number of scenarios increases. We do not observe a clear pattern in correlation with parameters β^w or $|K_s|$.

The profit of the system increases with the increase in β^w or in $|K_s|$ whereas it decreases with the increase in costs or in the number of scenarios (see Figure 4 and Tables A.2-A.9 in Appendix A).

We have already seen in Figures 5 and 8 in Section 4 that system tends to open more stations as we decrease cost values or increase β^w values. We do not observe any correlation with $|S|$ and although it seems that there might be a positive correlation with $|K_s|$, it is not very obvious.

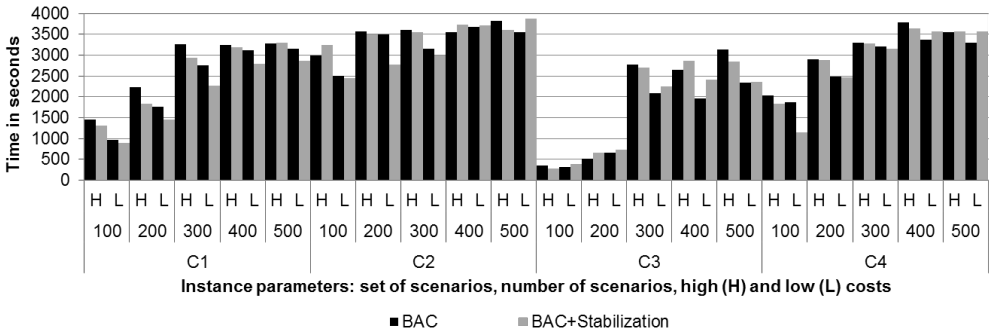
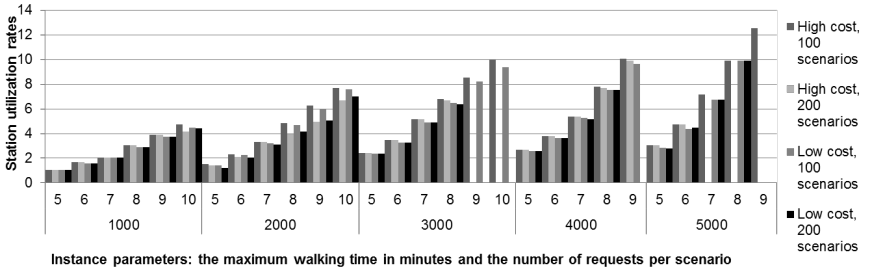
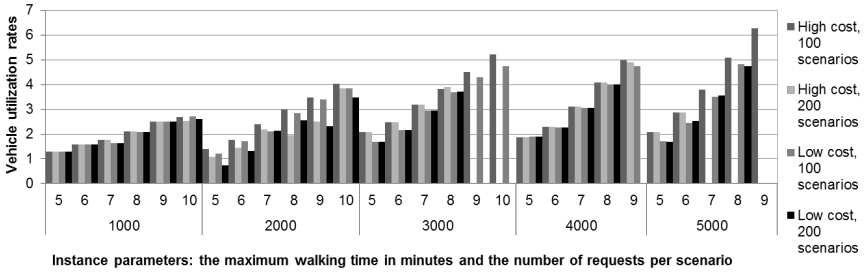


Figure 11: Average solving time (in seconds) of our algorithms for C1-C4



(a) The expected number of trips served per vehicle.



(b) The expected number of trips served per vehicle

Figure 12: Utilization rates for the stations and the vehicles.

As we have a multi-scenario problem, in order to provide an estimation of the utilization rates of the stations and the vehicles, we calculate the expected number of trips served for each instance via the formula $E[\overline{u}] = \sum_{s \in S} q_s \sum_{h \in \overline{H}_s} \overline{u}_h$. Then, we obtain the utilization rate for the stations as $E[\overline{u}]$ divided by the number of stations opened, and similarly, as $E[\overline{u}]$ divided by the number cars purchased for the vehicles. From Figure 12 of C3 instances, we observe that the utilization rates increase in parallel to the increase in β^w and especially, in $|K_s|$, whereas they decrease when $|S|$ or the $costfactor$ increases.

We provide a summary of our analysis in Table 1.

Finally, we do some additional experiments by using more variations of fixed costs for cars. These values are closer to the actual costs of Smart ED in the US. We still use a $costfactor = 10^5$ to divide the fixed costs of stations as otherwise the system does not provide a positive profit. We see from our experiments that the maximum profit that could be obtained among all instances falls below 5000 € for $\beta^w = 10$ and it is below 150 € for $\beta^w = 5$. As these values seems to be rather too small to run a profitable business, it

Table 1: Correlation between the changes on problem parameters and outputs

	$costfactor \uparrow$	$\beta^w \uparrow$	$\lvert S\rvert \uparrow$	$\lvert K_s\rvert \uparrow$
Obj	$\uparrow$	$\uparrow$	$\downarrow$	$\uparrow$
$\lvert\bar{J}\rvert$	$\uparrow$	mostly $\uparrow$	no pattern	no pattern
PP time	not applicable	$\uparrow$	no pattern	mostly $\uparrow$
PF time	decreases in average	mostly $\uparrow$	$\uparrow$	$\uparrow$
BAC time	decreases in average	mostly $\uparrow$	$\uparrow$	no pattern
BAC-S time	decreases in average	mostly $\uparrow$	$\uparrow$	no pattern
BAC Phase II	no pattern	no pattern	no pattern	no pattern
BAC-S Phase II	no pattern	no pattern	no pattern	no pattern
Avg. gap %	$\downarrow$	no pattern	mostly $\uparrow$	no pattern

might be necessary to find additional financial means to wave the fixed costs.

7. Conclusion

In this paper, we introduce a strategic decision-making problem arising in the design of a one-way station based electric car sharing system that operates under demand uncertainty. The strategic decisions include the number and location of the recharging stations which need to be selected meticulously as opening of these kind of facilities requires large amount of investment in money and time, and it is undesirable to replace them frequently. Our model takes into account the potential customer demand based on a relevant historical data and the expected revenue of the demand portion that could be served in a feasible way with the opened stations. As we aim to solve this problem optimally, we develop a mathematical formulation and a Benders decomposition algorithm that we further enrich with a stabilization procedure. We observe that our algorithm has the potential to solve real size problems to optimality, if not we can obtain very high quality solutions within very short amount of time.

We observe through our experimental study that the problem is harder to solve when the costs (station opening and car purchasing) are higher and as expected, less number of stations are operated and less profit is made. Moreover, even a small portion of increase in the maximum walking time for customers increases the profit significantly. Therefore, the accessibility measures should be selected very carefully when operating the system. The accessibility measures can also be used as a parameter in formation of pricing strategies. Another parameter that needs to be taken into account in design or operation of electric car sharing systems is the number of different demand scenarios. As we observe in this study, considering too few scenarios in taking decisions might cause an overestimation of the actual profit or misplacement of stations. The length of the planning horizon is also very effective in the profit of the system constructed. We observe that the profit increases as the number of requests per scenario increases, which can be interpreted as a longer planning period. A future study could focus on deciding the optimal length of the planning horizon for strategic decisions such as location of stations or recharging units.

The main focus of this work is on exact methods but the methods we propose can be easily modified or combined with practical procedures to obtain near optimal solutions.

As mentioned in Section 3, we can easily adapt our methods to more complicated variations of the problem we proposed here. These variations include considering time-dependent traveling time and energy consumption as well as visiting intermediate stations. An interesting and more challenging extension would

be to relax full battery restriction for availability and develop methods that can ensure sufficient battery at the beginning of each trip. In this case, an additional index for each operating car would be needed and the mathematical models might have problems in tackling large scale problems. On the other hand, some meta-heuristic methods might be developed to solve these problems heuristically.

Acknowledgements

This research is conducted under e4-share (Models for Ecological, Economical, Efficient, Electric Car-Sharing) project funded by FFG, INNOVIRIS and MIUR via JPI Urban Europe.

References

References

[1] Okan Arslan and Oya Ekin Karaşan. A Benders decomposition approach for the charging station location problem with plug-in hybrid electric vehicles. *Transportation Research Part B: Methodological*, 93:670–695, 2016.

[2] Johannes Asamer, Martin Reinthaler, Mario Ruthmair, Markus Straub, and Jakob Puchinger. Optimizing charging station locations for urban taxi providers. *Transportation Research Part A: Policy and Practice*, 85:233–246, 2016.

[3] Fouad Baouche, Romain Billot, Rochdi Trigui, and Nour-Eddin El Faouzi. Efficient allocation of electric vehicles charging stations: Optimization model and application to a dense urban network. *IEEE Intelligent Transportation Systems Magazine*, 6(3):33–43, 2014. ISSN 1939-1390. doi: 10.1109/mits.2014.2324023.

[4] Matthew Barth, Michael Todd, and Lei Xue. User-based vehicle relocation techniques for multiple-station shared-use vehicle systems. In *Transportation Research Board, 80th Annual Meeting*, 2004. URL `http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.360.8614`.

[5] Walid Ben-Ameur and José Neto. Acceleration of cutting-plane and column generation algorithms: Applications to network design. *Networks*, 49(1):3–17, 2007.

[6] Jacques F Benders. Partitioning procedures for solving mixed-variables programming problems. *Numerische Mathematik*, 4(1):238–252, 1962.

[7] Benjamin Biesinger, Bin Hu, Martin Stubenschrott, Ulrike Ritzinger, Matthias Prandtstetter. Optimizing charging station locations for electric car-sharing systems. *Evolutionary Computation in Combinatorial Optimization*, pages 157–172, 2017.

[8] John R Birge and François Louveaux. Two-stage recourse problems. In *Introduction to Stochastic Programming*, pages 181–263. Springer, 2011.

[9] Quentin Botton, Bernard Fortz, Luis Gouveia, and Michael Poss. Benders decomposition for the hop-constrained survivable network design problem. *INFORMS Journal on Computing*, 25(1):13–26, 2013.

[10] Burak Boyacı, Konstantinos G. Zografos, and Nikolas Geroliminis. An optimization framework for the development of efficient one-way car-sharing systems. *European Journal of Operational Research*, 240(3):718–733, February 2015. ISSN 0377-2217. doi: 10.1016/j.ejor.2014.07.020.

[11] Burak Boyacı, Konstantinos G Zografos, and Nikolas Geroliminis. An integrated optimization-simulation framework for vehicle and personnel relocations of electric carsharing systems with reservations. *Transportation Research Part B: Methodological*, 95:214–237, 2017.

[12] Georg Brandstätter, Claudio Gambella, Markus Leitner, Enrico Malaguti, Filippo Masini, Jakob Puchinger, Mario Ruthmair, and Daniele Vigo. Overview of optimization problems in electric car-sharing system design and management. In *Dynamic Perspectives on Managerial Decision Making*, pages 441–471. Springer, 2016.

[13] Georg Brandstätter, Markus Leitner, and Ivana Ljubić. Location of charging stations in electric car sharing systems. Technical report, Department of Statistics and Operations Research, University of Vienna, Vienna, Austria, 2016.

[14] Georg Brandstätter, Michael Kahr, and Markus Leitner. Determining optimal locations for charging stations of electric car-sharing systems under stochastic demand. *Transportation Research Part B: Methodological*, 104, pages 17–35, 2017.

[15] Maurizio Bruglieri, Alberto Colorni, and Alessandro Luè. The vehicle relocation problem for the one-way electric vehicle sharing: An application to the milan case. *Procedia - Social and Behavioral Sciences*, 111:18–27, February 2014. ISSN 1877-0428. doi: 10.1016/j.sbspro.2014.01.034.

[16] Hatice Çalık and Bernard Fortz. Location of stations in a one-way electric car sharing system. In *Computers and Communications (ISCC), 2017 IEEE Symposium on*, pages 134–139. IEEE, 2017.

[17] Hatice Çalık and Bernard Fortz. Demand forecasting models for one-way car sharing systems. Technical Report: Université Libre de Bruxelles, Brussels, Belgium, 2017.
https://dipot.ulb.ac.be/dspace/bitstream/2013/277634/3/Demand_Forecasting.pdf

[18] Hatice Calik, Markus Leitner, and Martin Luipersbeck. A Benders decomposition based framework for solving cable trench problems. *Computers & Operations Research*, 81:128–140, 2017.

[19] Aurélien Carlier, Alix Munier-Kordon, and Witold Klaudel. Mathematical model for the study of relocation strategies in one-way carsharing systems. *Transportation Research Procedia*, 10:374–383, 2015.

[20] Joana Cavadas, Gonçalo H.A. Correia, and Joao Gouveia. A mip model for locating slow-charging stations for electric vehicles in urban areas accounting for driver tours. *Transportation Research Part E: Logistics and Transportation Review*, 75:188–201, March 2015. ISSN 1366-5545. doi: 10.1016/j.tre.2014.11.005.

[21] T Donna Chen, Kara M Kockelman, Moby Khan, et al. The electric vehicle charging station location problem: a parking-based assignment method for seattle. In *92nd Annual Meeting of the Transportation Research Board. Washington DC, USA*, 2013. URL `http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.300.2883`.

[22] Monica Clemente, Maria Pia Fanti, Agostino M. Mangini, and Walter Ukovich. The vehicle relocation problem in car sharing systems: Modeling and simulation in a petri net framework. *Lecture Notes in Computer Science*, 7927:250–269, 2013.

[23] Gianni Codato and Matteo Fischetti. Combinatorial Benders' cuts for mixed-integer linear programming. *Operations Research*, 54(4):756–766, 2006.

[24] Gonçalo H.A. Correia and António Pais Antunes. Optimization approach to depot location and trip selection in one-way carsharing systems. *Transportation Research Part E: Logistics and Transportation Review*, 48(1):233–247, January 2012. ISSN 1366-5545. doi: 10.1016/j.tre.2011.06.003.

[25] Gonçalo H.A. Correia, Diana Ramos Jorge, and David Marques Antunes. The added value of accounting for users' flexibility and information on the potential of a station-based one-way car-sharing system: An application in Lisbon, Portugal. *Journal of Intelligent Transportation Systems*, 18(3):299–308, June 2014. ISSN 1547-2442. doi: 10.1080/15472450.2013.836928.

[26] Sevgi Erdoğan and Elise Miller-Hooks. A green vehicle routing problem. *Transportation Research Part E: Logistics and Transportation Review*, 48(1):100–114, January 2012. ISSN 1366-5545. doi: 10.1016/j.tre.2011.08.001.

[27] Ahmed El Fassi, Anjali Awasthi, and Marco Viviani. Evaluation of carsharing network's growth strategies through discrete event simulation. *Expert Systems with Applications*, 39(8):6692–6705, June 2012. ISSN 0957-4174. doi: 10.1016/j.eswa.2011.11.071.

[28] Ángel Felipe, M. Teresa Ortuño, Giovanni Righini, and Gregorio Tirado. A heuristic approach for the green vehicle routing problem with multiple technologies and partial recharges. *Transportation Research Part E: Logistics and Transportation Review*, 71(0):111–128, 2014. ISSN 1366-5545. doi: 10.1016/j.tre.2014.09.003.

[29] Matteo Fischetti, Ivana Ljubić, and Markus Sinnl. Redesigning Benders decomposition for large-scale facility location. *Management Science*, 2016.

[30] Bernard Fortz and Michael Poss. An improved Benders decomposition applied to a multi-layer network design problem. *Operations Research Letters*, 37(5):359–364, 2009.

[31] Bernard Fortz and Mikkel Thorup. Increasing internet capacity using local search. *Computational Optimization and Applications*, 29(1):13–48, 2004.

[32] Inês Frade, Anabela Ribeiro, Gonçalo Gonçalves, and António Pais Antunes. Optimal location of charging stations for electric vehicles in a neighborhood in lisbon, Portugal. *Transportation Research Record: Journal of the Transportation Research Board*, 2252:91–98, December 2011. ISSN 0361-1981. doi: 10.3141/2252-12.

[33] Claudio Gambella, Enrico Malaguti, Filippo Masini, and Daniele Vigo. Optimizing relocation operations in electric car-sharing. *Omega*, 81:234–245, 2018.

[34] Shaoyun Ge, Liang Feng, and Hong Liu. The planning of electric vehicle charging station based on grid partition method. In *2011 International Conference on Electrical and Control Engineering (ICECE)*. IEEE, September 2011. ISBN http://id.crossref.org/isbn/978-1-4244-8162-0. doi: 10.1109/iceceng.2011.6057636.

[35] Sascha Herrmann, Frederik Schulte, and Stefan Voß. Increasing acceptance of free-floating car sharing systems using smart relocation strategies: a survey based study of car2go hamburg. In *Computational Logistics*, pages 151–162. Springer, 2014.

[36] Andrea Hess, Francesco Malandrino, Moritz Bastian Reinhardt, Claudio Casetti, Karin Anna Hummel, and Jose M. Barceló-Ordinas. Optimal deployment of charging stations for electric vehicular networks. In *Proceedings of the First*

Workshop on Urban Networking, UrbaNe '12, pages 1–6, New York, NY, USA, 2012. ACM. ISBN 978-1-4503-1781-8. doi: 10.1145/2413236.2413238.

[37] John N Hooker and Greger Ottosson. Logic-based benders decomposition. *Mathematical Programming*, 96(1):33–60, 2003.

[38] Diana Jorge, Goncalo H.A. Correia, and Cynthia Barnhart. Comparing optimal relocation operations with simulated relocation policies in one-way carsharing systems. *IEEE Transactions on Intelligent Transportation Systems*, 15(4): 1667–1675, August 2014. ISSN 1558-0016. doi: 10.1109/tits.2014.2304358.

[39] Alvina G.H. Kek, Ruey Long Cheu, Qiang Meng, and Chau Ha Fung. A decision support system for vehicle relocation operations in carsharing systems. *Transportation Research Part E: Logistics and Transportation Review*, 45(1):149–158, January 2009. ISSN 1366-5545. doi: 10.1016/j.tre.2008.02.008.

[40] Xiaopeng Li, Jiaqi Ma, Jianxun Cui, Amir Ghiasi, and Fang Zhou. Design framework of large-scale one-way electric vehicle sharing systems: A continuum approximation model. *Transportation Research Part B: Methodological*, 88:21–45, 2016.

[41] Magnanti, T.L. and Wong, R.T., 1981. Accelerating Benders decomposition: Algorithmic enhancement and model selection criteria. *Operations research*, 29(3), pp.464–484, 1981.

[42] Eduardo Álvarez Miranda, Ivana Ljubić, Martin Luipersbeck, and Markus Sinnl. Solving minimum-cost shared arborescence problems. *European Journal of Operational Research*, 258(3):887 – 901, 2017.

[43] Samuel Pelletier, Ola Jabali, and Gilbert Laporte. 50th anniversary invited article—goods distribution with electric vehicles: Review and research perspectives. *Transportation Science*, 50(1):3–22, 2016. doi: 10.1287/trsc.2015.0646.

[44] Mario Ruthmair, Martin Stubenschrott. Input Data Processing. Technical Report: AIT Austrian Institute of Technology, Vienna, Austria, 2015.

[45] Michael Schneider, Andreas Stenger, and Dominik Goeke. The electric vehicle-routing problem with time windows and recharging stations. *Transportation Science*, 48(4):500–520, November 2014. ISSN 1526-5447. doi: 10.1287/trsc.2013.0490.

[46] Frederik Schulte and Stefan Voß. Decision support for environmental-friendly vehicle relocations in free-floating car sharing systems: The case of car2go. *Procedia CIRP*, 30:275–280, 2015.

[47] Hengsong Wang, Qi Huang, Changhua Zhang, and Aihua Xia. A novel approach for the layout of electric vehicle charging station. In *The 2010 International Conference on Apperceiving Computing and Intelligence Analysis Proceeding*. IEEE, December 2010. ISBN http://id.crossref.org/isbn/978-1-4244-8025-8. doi: 10.1109/icacia.2010.5709852.

[48] Ying-Wei Wang and Chuah-Chih Lin. Locating multiple types of recharging stations for battery-powered electric vehicle transport. *Transportation Research Part E: Logistics and Transportation Review*, 58:76–87, November 2013. ISSN 1366-5545. doi: 10.1016/j.tre.2013.07.003.

[49] Chris Whong. NYC Taxi Trip Data `https://chriswhong.com/`, 2018.

Appendix A. Detailed computational results

Table A.2: Model results for $|S| = 100$ and $costfactor = 10^5$ - gaps & solution values

		LP		RPF		PF		
$\|K_s\|$	β^w	Obj	g1(%)	Obj	g2(%)	Obj	g2(%)	time
1000	5	7342.31	1.71	7218.65	0.00	7218.65	0.00	4.12
1000	6	15605.92	3.88	15022.45	0.00	15022.45	0.00	15.76
1000	7	27573.03	1.80	27085.46	0.00	27085.46	0.00	46.71
1000	8	37518.81	1.45	36980.87	0.00	36980.87	0.00	247.10
1000	9	44984.52	0.58	44723.11	0.00	44723.11	0.00	656.09
1000	10	52304.87	0.28	52158.64	0.00	52158.64	0.00	723.22
1000	15	89960.52	0.13	0.00	NA	0.00	NA	TL
2000	5	21016.22	0.85	20839.93	0.00	20839.93	0.00	11.72
2000	6	32538.52	1.90	31931.30	0.00	31931.30	0.00	33.56
2000	7	45264.42	0.80	44907.15	0.00	44907.15	0.00	97.92
2000	8	53763.37	0.32	53593.97	0.00	53593.97	0.00	294.15
2000	9	61039.66	0.06	61004.03	0.00	61004.03	0.00	850.56
2000	10	68256.41	0.12	68173.08	0.00	68173.08	0.00	2987.40
2000	15	120772.52	0.05	0.00	NA	NA	NA	TL
3000	5	25213.00	1.74	24781.00	0.00	24781.00	0.00	18.78
3000	6	42570.33	0.87	42204.00	0.00	42204.00	0.00	86.56
3000	7	53092.93	0.24	52964.00	0.00	52964.00	0.00	794.61
3000	8	63073.87	0.22	62933.00	0.00	62933.00	0.00	784.47
3000	9	72583.38	0.15	72480.00	0.00	72476.00	0.04	TL
3000	10	78374.00	0.02	78358.00	0.00	78358.00	0.00	TL
3000	15	141114.40	NA	0.00	NA	NA	NA	TL
4000	5	32185.67	0.64	31980.00	0.00	31980.00	0.00	27.40
4000	6	50607.83	0.32	50444.00	0.00	50444.00	0.00	94.88
4000	7	63640.95	0.25	63485.00	0.00	63485.00	0.00	1720.41
4000	8	70676.08	0.14	70577.00	0.00	70577.00	0.00	2042.30
4000	9	76461.20	0.04	76346.00	0.12	76431.00	0.01	1655.19
4000	10	79728.28	0.03	79666.00	0.00	79708.00	0.00	2329.47
4000	15	149452.50	NA	0.00	NA	NA	NA	TL
5000	5	38549.29	0.99	38169.82	0.00	38169.82	0.00	68.72
5000	6	53705.84	0.41	53488.83	0.00	53488.83	0.00	312.48
5000	7	65663.05	0.17	65554.10	0.00	65554.10	0.00	1877.70
5000	8	72589.00	0.05	72554.02	0.00	72554.02	0.00	2357.71
5000	9	79130.76	0.16	79005.40	0.00	79005.40	0.04	TL
5000	10	85073.95	0.09	84986.33	0.00	84998.90	0.00	TL
5000	15	161011.19	NA	NA	NA	NA	NA	TL
Avg.:			0.64		0.00		0.00	1518.65
Max:			3.88		0.12		0.04	3777.61

Table A.3: Model results with $|S| = 100$ and $costfactor = 10^6$ - gaps & solution values

		LP		RPF		PF		
$\|K_s\|$	β^w	Obj	g1(%)	Obj	g2(%)	Obj	g2(%)	time
1000	5	16980.75	0.03	16975.83	0.00	16975.83	0.00	5.59
1000	6	27698.48	0.08	27675.73	0.00	27675.73	0.00	14.35
1000	7	41089.05	0.14	41030.45	0.00	41030.45	0.00	42.35
1000	8	51404.74	0.05	51380.12	0.00	51380.12	0.00	116.04
1000	9	58869.93	0.05	58838.80	0.00	58838.80	0.00	543.88
1000	10	66417.96	0.02	66402.34	0.00	66402.34	0.00	521.08
1000	15	106736.62	0.00	0.00	NA	0.00	NA	TL
2000	5	33364.02	0.04	33349.13	0.00	33349.13	0.00	15.77
2000	6	45939.10	0.07	45904.90	0.00	45904.90	0.00	41.48
2000	7	59443.25	0.07	59399.62	0.00	59399.62	0.00	139.59
2000	8	67929.84	0.04	67903.56	0.00	67903.56	0.00	1096.80
2000	9	75852.91	0.01	75843.23	0.00	75843.23	0.00	TL
2000	10	83831.89	0.01	83820.27	0.00	83820.27	0.00	TL
2000	15	139678.53	0.00	0.00	NA	NA	NA	TL
3000	5	36036.00	0.09	36003.00	0.00	36003.00	0.00	23.38
3000	6	55492.75	0.05	55466.00	0.00	55466.00	0.00	96.77
3000	7	67553.75	0.04	67527.00	0.00	67527.00	0.00	248.83
3000	8	78213.28	0.01	78209.00	0.00	78209.00	0.00	701.07
3000	9	88457.67	0.01	88446.00	0.00	88446.00	0.00	2693.32
3000	10	94913.75	0.01	94909.00	0.00	94909.00	0.00	2726.03
3000	15	160490.00	NA	0.00	NA	NA	NA	TL
4000	5	45154.33	0.04	45137.00	0.00	45137.00	0.00	32.39
4000	6	65262.25	0.02	65252.00	0.00	65252.00	0.00	115.26
4000	7	79767.07	0.02	79753.00	0.00	79753.00	0.00	1746.54
4000	8	86871.00	0.00	86868.00	0.00	86868.00	0.00	1035.95
4000	9	93277.00	0.00	93277.00	0.00	93277.00	0.00	1655.19
4000	10	96947.00	0.00	96947.00	0.00	96947.00	0.00	1824.23
4000	15	168970.00	NA	0.00	NA	NA	NA	TL
5000	5	51641.00	0.05	51614.58	0.00	51614.58	0.00	53.21
5000	6	68400.66	0.06	68362.75	0.00	68362.75	0.00	423.10
5000	7	81719.65	0.03	81695.37	0.00	81695.37	0.00	1223.83
5000	8	89801.18	0.02	89786.21	0.00	89786.21	0.00	2706.08
5000	9	96651.38	0.01	96645.94	0.00	96645.70	0.00	3337.07
5000	10	103253.62	0.00	103252.48	0.00	103252.48	0.00	3024.76
5000	15	180913.30	NA	0.00	NA	NA	NA	TL
Avg.:			0.03		0.00		0.00	1491.68
Max:			0.14		0.00		0.00	3888.52

Table A.4: Model results for $|S| = 200$ and $costfactor = 10^5$ - gaps & solution values

		LP		RPF		PF		
$\|K_s\|$	β^w	Obj	g1(%)	Obj	g2(%)	Obj	g2(%)	time
1000	5	6969.82	2.00	6832.88	0.00	6832.88	0.00	14.46
1000	6	15444.12	3.74	14887.28	0.00	14887.28	0.00	62.97
1000	7	27442.90	1.66	26994.52	0.00	26994.52	0.00	265.89
1000	8	37319.10	1.50	36740.66	0.18	36766.38	0.00	2550.04
1000	9	44926.41	0.64	0.00	NA	44496.02	0.38	TL
1000	10	52354.37	0.30	0.00	NA	0.00	NA	TL
1000	15	0.02	NA	0.00	NA	NA	NA	TL
2000	5	17369.70	0.69	17251.00	0.00	17251.00	0.00	31.43
2000	6	32614.25	1.22	32220.00	0.00	32220.00	0.00	147.28
2000	7	44027.19	1.16	43229.00	0.74	43524.00	0.09	TL
2000	8	52368.69	0.14	0.00	NA	52294.00	0.00	2088.65
2000	9	62119.02	1.55	0.00	NA	0.00	NA	TL
2000	10	68569.30	1.74	0.00	NA	0.00	NA	TL
2000	15	0.00	NA	0.00	NA	NA	NA	TL
3000	5	25213.00	1.74	24781.00	0.00	24781.00	0.00	63.67
3000	6	42570.33	0.87	42204.00	0.00	42204.00	0.00	485.12
3000	7	53092.93	0.24	52964.00	0.00	52964.00	0.00	3286.40
3000	8	63073.87	2.76	0.00	NA	0.00	NA	TL
3000	9	72583.38	1.42	0.00	NA	0.00	NA	TL
3000	10	78374.00	0.75	0.00	NA	0.00	NA	TL
4000	5	32046.28	0.64	31842.00	0.00	31842.00	0.00	160.84
4000	6	50500.08	0.32	50337.72	0.00	50337.72	0.00	575.42
4000	7	63537.28	0.25	63005.95	0.62	0.00	NA	TL
4000	8	70579.57	0.14	0.00	NA	0.00	NA	TL
4000	9	76396.01	0.03	0.00	NA	0.00	NA	TL
4000	10	79660.24	0.05	0.00	NA	NA	NA	TL
5000	5	38280.05	0.91	37934.12	0.00	37934.12	0.00	673.09
5000	6	53494.06	0.36	53216.43	0.16	53300.98	0.00	1903.53
5000	7	65459.66	0.15	0.00	NA	0.00	NA	TL
5000	8	72521.60	0.05	0.00	NA	0.00	NA	TL
5000	9	79074.33	0.16	0.00	NA	0.00	NA	TL
5000	10	85072.10	0.09	0.00	NA	NA	NA	TL
Avg.:			0.91		0.11		0.03	2438.34
Max:			3.74		0.74		0.38	3739.77

Table A.5: Model results with $|S| = 200$ and $costfactor = 10^6$ - gaps & solution values

		LP		RPF		PF		
$\|K_s\|$	β^w	Obj	g1(%)	Obj	g2(%)	Obj	g2(%)	time
1000	5	16766.10	0.00	16766.10	0.00	16766.10	0.00	12.64
1000	6	27615.35	0.07	27597.19	0.00	27597.19	0.00	30.69
1000	7	41037.19	0.11	40992.58	0.00	40992.58	0.00	79.90
1000	8	51395.72	0.05	51371.33	0.00	51371.33	0.00	354.34
1000	9	58943.52	0.07	58855.20	0.11	58899.58	0.01	TL
1000	10	66533.64	0.01	66524.12	0.01	66528.32	0.00	1782.33
1000	15	0.21	NA	0.00	NA	NA	NA	TL
2000	5	27767.50	0.03	27758.00	0.00	27758.00	0.00	24.26
2000	6	44523.50	0.09	44485.00	0.00	44485.00	0.00	89.46
2000	7	56945.73	0.04	56921.00	0.00	56921.00	0.00	1051.88
2000	8	65833.38	0.01	65824.00	0.00	65824.00	0.00	TL
2000	9	76688.18	0.80	0.00	NA	0.00	NA	TL
2000	10	83826.47	0.21	0.00	NA	0.00	NA	TL
2000	15	0.01	NA	0.00	NA	NA	NA	TL
3000	5	36036.00	0.09	36003.00	0.00	36003.00	0.00	134.59
3000	6	55492.75	0.05	55466.00	0.00	55466.00	0.00	388.46
3000	7	67553.75	0.04	67527.00	0.00	67527.00	0.00	1150.57
3000	8	78213.28	0.01	78206.00	0.00	78209.00	0.00	1877.87
3000	9	88457.67	0.16	0.00	NA	0.00	NA	TL
3000	10	94913.75	0.22	0.00	NA	0.00	NA	TL
4000	5	45063.29	0.04	45047.24	0.00	45047.24	0.00	177.75
4000	6	65174.09	0.02	65164.01	0.00	65164.01	0.00	747.65
4000	7	79657.46	0.02	79643.34	0.00	79643.34	0.01	TL
4000	8	86775.66	0.01	86768.81	0.00	0.00	NA	TL
4000	9	93220.05	0.00	93220.05	0.00	0.00	NA	TL
4000	10	96908.38	0.01	0.00	NA	0.00	NA	TL
5000	5	51525.43	0.03	51508.09	0.00	51508.09	0.00	429.57
5000	6	68232.66	0.03	68215.50	0.00	68215.50	0.00	1099.12
5000	7	81548.30	0.03	81521.92	0.00	81521.92	0.00	2349.23
5000	8	89735.76	0.02	89720.81	0.00	89714.20	0.02	TL
5000	9	96601.81	0.00	0.00	NA	NA	NA	TL
5000	10	103249.59	0.00	0.00	NA	0.00	NA	TL
Avg.:			0.07		0.01		0.00	2081.68
Max:			0.80		0.11		0.02	3768.34

Table A.6: Comparison of PF with BAC for $|S| = 100$ and $costfactor = 10^5$

		PF			BAC		BAC Phase I			BAC Phase II			
$\|K_s\|$	β^w	g1(%)	time	nodes	g1(%)	time	lazy	user	nodes	lazy	user	nodes	$\|\overline{J}\|$
1000	5	0.00	4.12	0	0.00	36.44	24	44	17				49
1000	6	0.00	15.76	0	0.00	95.51	42	82	104				62
1000	7	0.00	46.71	4	0.00	80.21	15	63	3				67
1000	8	0.00	247.10	500	0.00	1518.94	148	87	27853				67
1000	9	0.00	656.09	40	0.00	2826.21	21	60	241	6	53	197	66
1000	10	0.00	723.22	83	0.00	498.66	19	53	328				66
1000	15	100.00	3629.30	0	0.00	2942.05	8	19	15	6	0	0	76
2000	5	0.00	11.72	0	0.00	6.96	4	1	0				60
2000	6	0.00	33.56	0	0.00	18.87	4	1	0				67
2000	7	0.00	97.92	0	0.00	25.42	2	0	0				72
2000	8	0.00	294.15	0	0.00	52.37	4	1	0				68
2000	9	0.00	850.56	0	0.00	120.18	4	1	0				70
2000	10	0.00	2987.40	1068	0.00	604.06	9	7	13	3	0	0	72
2000	15	NA	TL	NA	0.00	1763.26	1	6	0				81
3000	5	0.00	18.78	0	0.81	TL	130	353	58812	165	1040	8146	58
3000	6	0.00	86.56	0	0.40	TL	134	634	7781				66
3000	7	0.00	794.61	230	0.19	TL	51	493	0				67
3000	8	0.00	784.47	20	1.96	TL	16	84	0				69
3000	9	0.00	TL	511	1.90	TL	11	81	0				74
3000	10	0.00	TL	969	0.00	3481.80	21	84	985				76
3000	15	NA	TL	NA	NA	TL	1	4	0				NA
4000	5	0.00	27.40	0	0.00	11.66	4	1	0				64
4000	6	0.00	94.88	0	0.00	22.17	4	1	0				68
4000	7	0.00	1720.41	1111	0.00	184.13	21	6	64				74
4000	8	0.00	2042.30	525	0.00	262.33	10	6	47				74
4000	9	0.00	1655.19	967	0.00	385.09	4	7	5				75
4000	10	0.00	2329.47	99	0.00	636.52	6	7	3				76
4000	15	NA	TL	NA	NA	TL	0	6	0				NA
5000	5	0.00	68.72	36	0.00	62.78	23	38	75				66
5000	6	0.00	312.48	66	0.00	106.57	14	22	80				69
5000	7	0.00	1877.70	1177	0.00	359.93	27	20	354				75
5000	8	0.00	2357.71	2052	0.00	481.01	10	13	35				77
5000	9	0.00	TL	942	0.00	637.82	6	11	22				78
5000	10	0.00	TL	16	0.00	1087.72	13	8	60				79
5000	15	NA	TL	NA	NA	TL	NA	NA	NA				NA
Avg.:		3.23	1518.65		0.16	1460.36							
Max:		100.00	3777.61		1.96	7208.06							

Table A.7: Comparison of PF with BAC for $|S| = 100$ and $costfactor = 10^6$

		PF			BAC		BAC Phase I			BAC Phase II			
$\|K_s\|$	β^w	g1(%)	time	nodes	g1(%)	time	lazy	user	nodes	lazy	user	nodes	$\|\overline{J}\|$
1000	5	0.00	5.59	0	0.00	9.70	19	22	0				68
1000	6	0.00	14.35	0	0.00	14.74	19	27	0				74
1000	7	0.00	42.35	2	0.00	35.82	34	44	44				76
1000	8	0.00	116.04	11	0.00	49.91	9	23	10				77
1000	9	0.00	543.88	14	0.00	90.99	10	41	24				73
1000	10	0.00	521.08	0	0.00	184.31	15	24	10				73
1000	15	100.00	TL	0	0.00	1691.83	6	17	0	6	0	0	80
2000	5	0.00	15.77	0	0.00	7.06	3	0	0				71
2000	6	0.00	41.48	0	0.00	14.47	4	1	0				75
2000	7	0.00	139.59	0	0.00	24.15	6	2	0				78
2000	8	0.00	1096.80	1557	0.00	63.43	10	5	10				76
2000	9	0.00	TL	75	0.00	134.13	6	7	9				76
2000	10	0.00	TL	858	0.00	284.24	8	6	56				77
2000	15	NA	TL	NA	0.00	2633.47	3	10	0				NA
3000	5	0.00	23.38	0	0.00	962.87	60	1004	179				64
3000	6	0.00	96.77	0	0.00	1371.60	50	765	331				69
3000	7	0.00	248.83	0	0.00	1876.95	50	695	1084				75
3000	8	0.00	701.07	0	0.03	TL	42	368	0				76
3000	9	0.00	2693.32	0	0.01	TL	17	219	0				77
3000	10	0.00	2726.03	24	0.00	TL	21	145	62				80
3000	15	NA	TL	NA	NA	TL	1	4	0				NA
4000	5	0.00	32.39	0	0.00	9.56	3	0	0				71
4000	6	0.00	115.26	0	0.00	21.85	3	1	0				76
4000	7	0.00	1746.54	1017	0.00	70.61	5	7	18				77
4000	8	0.00	1035.95	0	0.00	177.57	8	8	0				77
4000	9	0.00	1655.19	0	0.00	218.38	3	2	0				79
4000	10	0.00	1824.23	0	0.00	300.02	1	2	0				79
4000	15	NA	TL	NA	NA	TL	0	5	0				NA
5000	5	0.00	53.21	0	0.00	53.51	22	82	50				75
5000	6	0.00	423.10	90	0.00	82.80	21	28	24				76
5000	7	0.00	1223.83	17	0.00	141.18	14	16	36				78
5000	8	0.00	2706.08	546	0.00	201.82	2	8	55				80
5000	9	0.00	3337.07	28	0.00	492.26	11	7	0	5	0	0	80
5000	10	0.00	3024.76	0	0.00	437.84	2	1	0				81
5000	15	NA	TL	NA	NA	TL	0	0	0				NA
Avg.:		3.23	1485.64		0.00	959.53							
Max:		100.00	3888.52		0.03	3699.71							

Table A.8: Comparison of PF with BAC for $|S| = 200$ and $costfactor = 10^5$

		PF			BAC		BAC Phase I			BAC Phase II			
$\|K_s\|$	β^w	g1(%)	time	nodes	g1(%)	time	lazy	user	nodes	lazy	user	nodes	$\|\overline{J}\|$
1000	5	0.00	14.46	0	0.00	52.38	31	35	195				48
1000	6	0.00	62.97	0	0.00	268.96	31	70	206	9	0	0	62
1000	7	0.00	265.89	3	0.00	155.95	25	46	11	10	0	0	68
1000	8	0.00	2550.04	1079	0.00	2606.79	98	76	21161	11	0	0	66
1000	9	0.32	TL	3	0.00	969.98	46	56	1194				66
1000	10	100.00	TL	0	0.00	866.58	18	45	581				66
1000	15	NA	TL	0	NA	TL	0	12	0				NA
2000	5	0.00	31.43	0	0.00	1280.23	51	417	489				53
2000	6	0.00	147.28	0	1.52	TL	58	569	0				62
2000	7	0.00	TL	173	0.44	TL	133	974	1103				66
2000	8	0.00	2088.65	0	0.21	TL	43	589	10				64
2000	9	100.00	TL	0	0.00	TL	21	203	0				71
2000	10	100.00	TL	0	0.00	TL	13	53	0				70
2000	15	NA	TL	0	NA	TL	1	3	0				NA
3000	5	0.00	63.67	0	5.26	TL	170	368	23932	61	1186	0	58
3000	6	0.00	485.12	0	0.81	TL	42	439	0				66
3000	7	0.00	TL	144	2.75	TL	30	190	0				67
3000	8	100.00	TL	0	0.00	TL	14	63	0				71
3000	9	100.00	TL	0	0.00	TL	12	51	0				72
3000	10	100.00	TL	0	0.00	TL	7	36	0				76
4000	5	0.00	160.84	0	0.00	27.79	5	2	0				64
4000	6	0.00	575.42	0	0.00	45.32	3	3	0				68
4000	7	100.00	TL	0	0.00	264.77	17	8	47				74
4000	8	100.00	TL	0	0.00	335.08	5	6	86				74
4000	9	100.00	TL	0	0.00	3494.47	6	5	7	4	7	51	75
4000	10	NA	TL	0	0.00	867.07	3	6	71				76
5000	5	0.00	673.09	25	0.00	69.24	14	30	92				66
5000	6	0.00	1903.53	0	0.00	264.90	14	25	50				69
5000	7	100.00	TL	0	0.00	463.73	17	21	151				75
5000	8	100.00	TL	0	0.00	TL	0	11	37	10	12	9	75
5000	9	100.00	TL	0	0.00	813.33	6	9	35				77
5000	10	NA	TL	0	0.00	TL	0	6	11	8	9	0	78
Avg.:		42.87	2438.34		0.37	2236.12							
Max:		100.00	3739.77		5.26	7207.33							

Table A.9: Comparison of PF with BAC for $|S| = 200$ and $costfactor = 10^6$

		PF			BAC		BAC Phase I			BAC Phase II			
$\|K_s\|$	β^w	g1(%)	time	nodes	g1(%)	time	lazy	user	nodes	lazy	user	nodes	$\|\overline{J}\|$
1000	5	0.00	12.64	0	0.00	9.75	18	14	0				68
1000	6	0.00	30.69	0	0.00	61.61	36	42	87	8	0	0	75
1000	7	0.00	79.90	0	0.00	66.85	28	28	13	9	0	0	77
1000	8	0.00	354.34	0	0.00	209.72	17	24	56	10	0	0	77
1000	9	0.01	TL	0	0.00	TL	26	51	91200	18	29	7044	76
1000	10	0.00	1782.33	0	0.00	484.39	7	25	45				74
1000	15	NA	TL	0	0.00	TL	9	13	158	15	0	0	80
2000	5	0.00	24.26	0	0.00	804.65	56	641	83				56
2000	6	0.00	89.46	0	0.00	2546.39	57	1520	416				64
2000	7	0.00	1051.88	0	0.00	2201.05	47	861	243				68
2000	8	0.00	TL	147	0.00	3081.48	45	626	333				68
2000	9	100.00	TL	0	0.00	TL	18	101	0				73
2000	10	100.00	TL	0	0.00	TL	25	109	0				76
2000	15	NA	TL	0	NA	TL	1	4	0				NA
3000	5	0.00	134.59	0	0.00	1719.39	50	890	84				64
3000	6	0.00	388.46	0	0.00	2744.97	43	856	314				69
3000	7	0.00	1150.57	7	0.00	3051.89	45	553	2983				74
3000	8	0.00	1877.87	0	0.05	TL	40	302	0				76
3000	9	100.00	TL	0	0.00	TL	22	121	0				79
3000	10	100.00	TL	0	0.00	TL	13	44	0				83
4000	5	0.00	177.75	0	0.00	74.96	32	0	184				71
4000	6	0.00	747.65	0	0.00	66.64	5	9	0				76
4000	7	0.00	TL	361	0.00	138.63	7	8	25				77
4000	8	100.00	TL	0	0.00	306.16	5	9	38				78
4000	9	100.00	TL	0	0.00	343.61	3	3	0				80
4000	10	100.00	TL	0	0.00	500.06	2	2	0				79
5000	5	0.00	429.57	153	0.00	154.87	32	64	22				77
5000	6	0.00	1099.12	0	0.00	730.80	14	29	40	13	34	123	76
5000	7	0.00	2349.23	180	0.00	488.06	19	13	117	10	0	0	79
5000	8	0.01	TL	0	0.00	756.91	8	8	36				80
5000	9	NA	TL	0	0.00	922.29	3	3	0	10	0	0	80
5000	10	NA	TL	0	0.00	1404.04	3	3	0				81
Avg.:		25.00	2081.68		0.00	1758.44							
Max:		100.00	3768.34		0.05	7226.77							

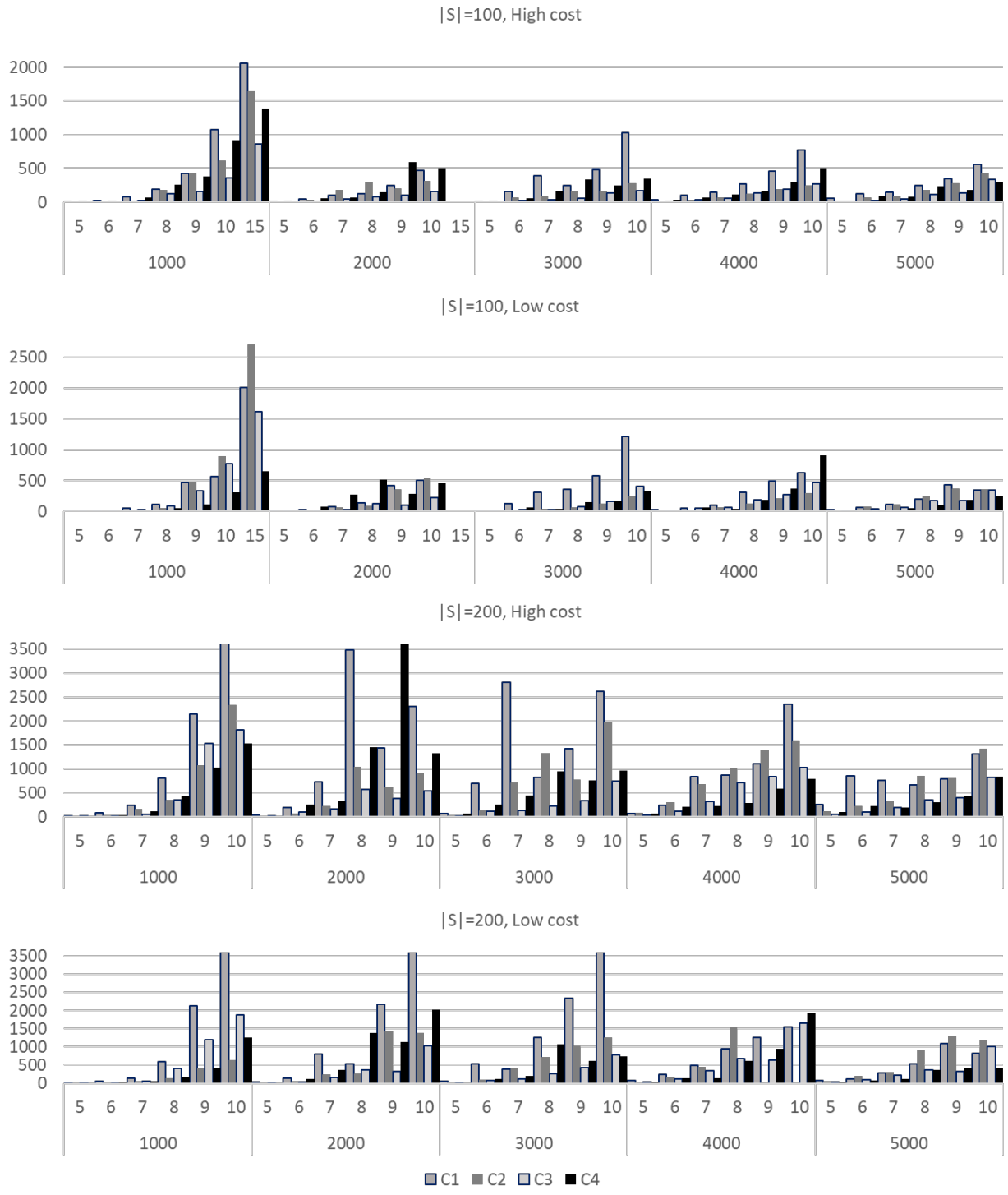


Figure A.13: LP solving times per instance for C1-C4

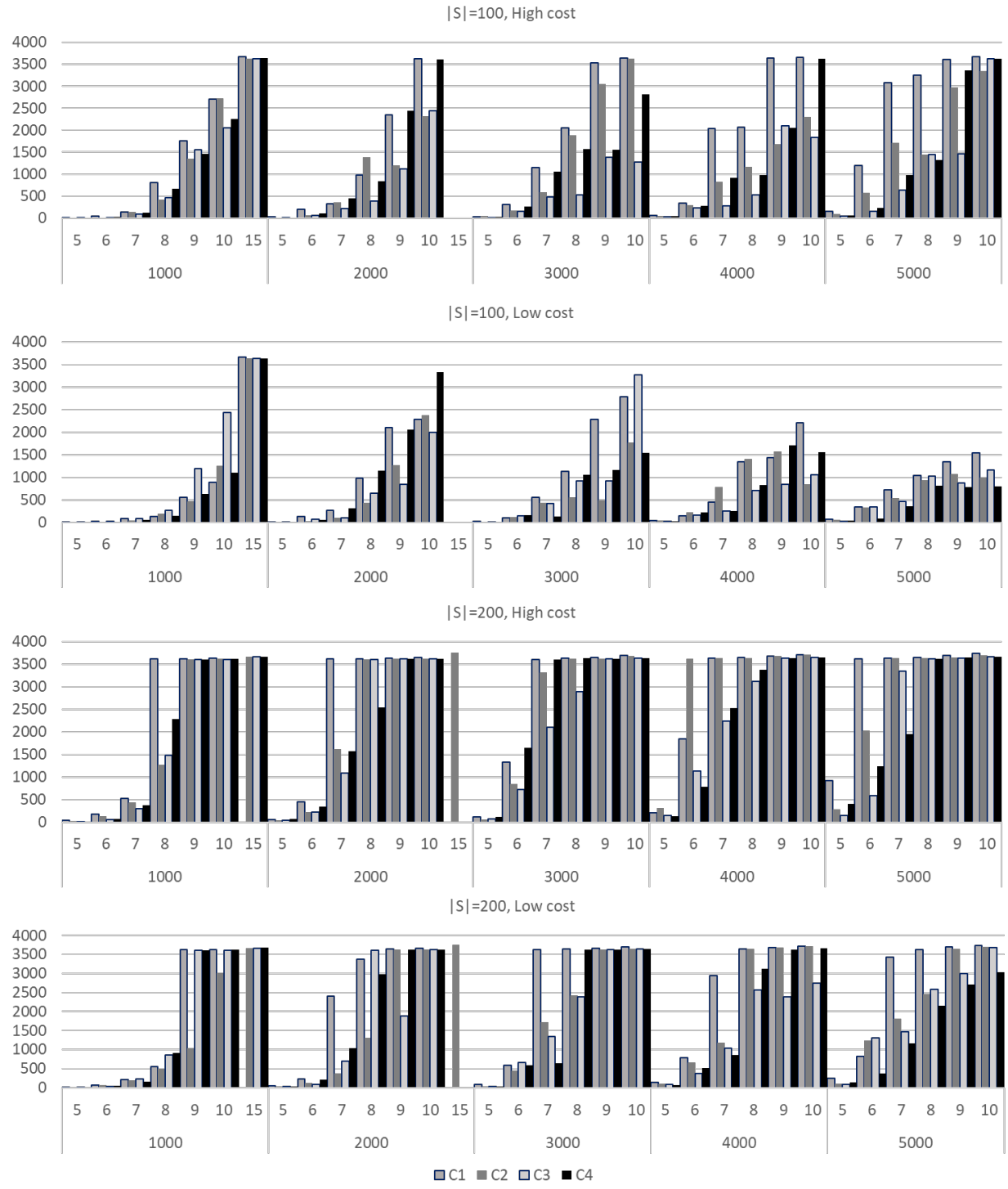


Figure A.14: RPF solving times per instance for C1-C4

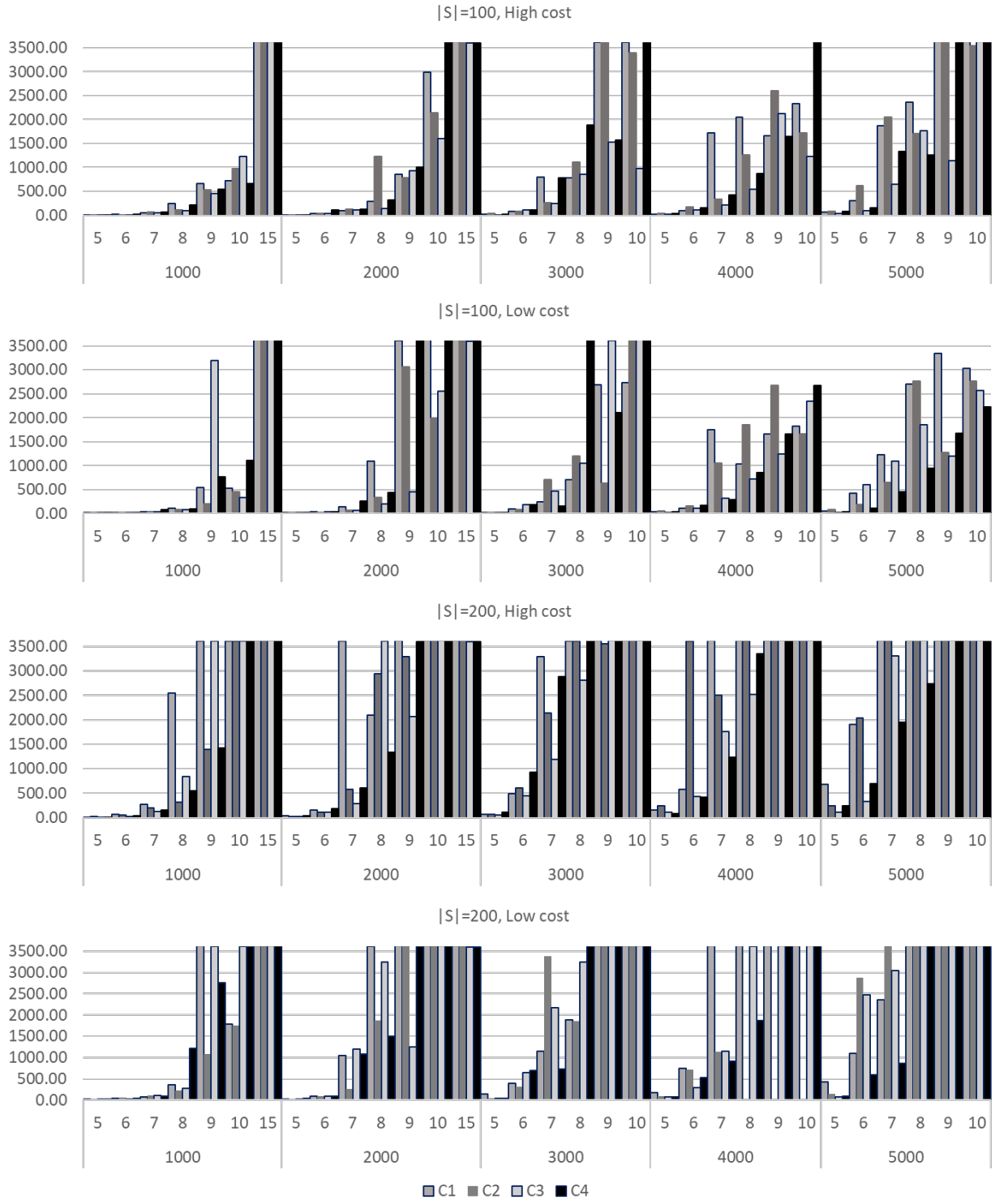


Figure A.15: PF solving times per instance for C1-C4

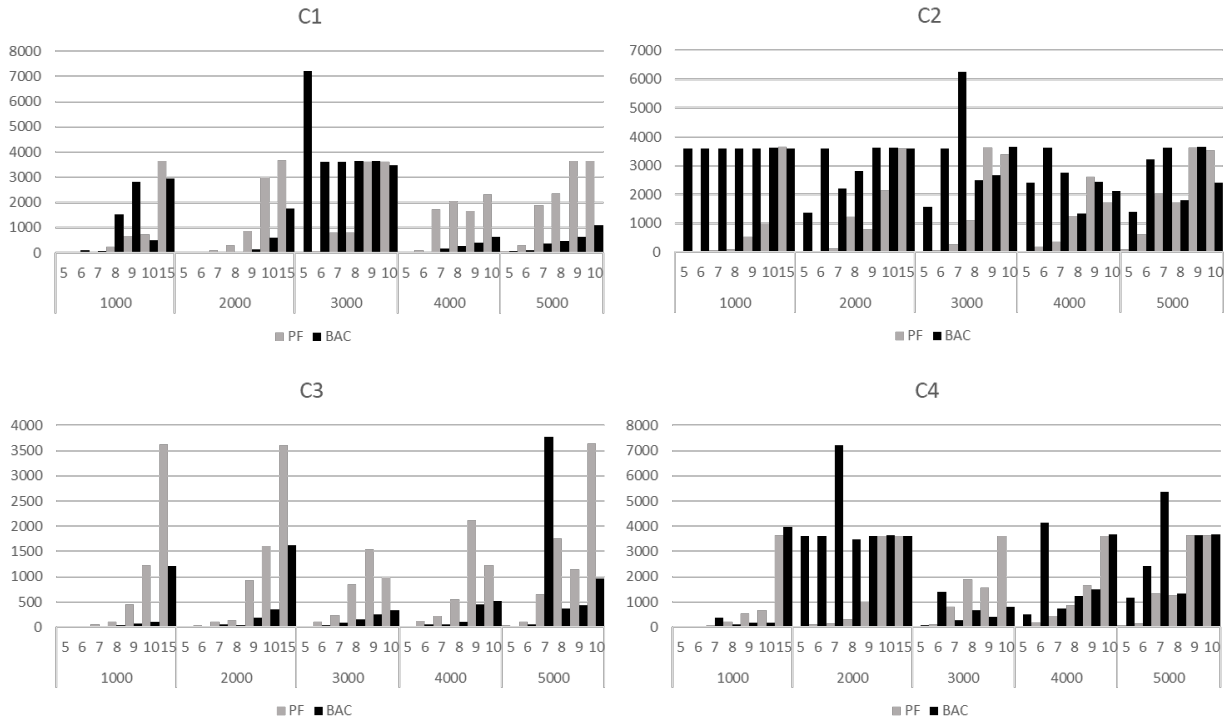


Figure A.16: Solving time of PF and BAC for $|S| = 100$, High cost

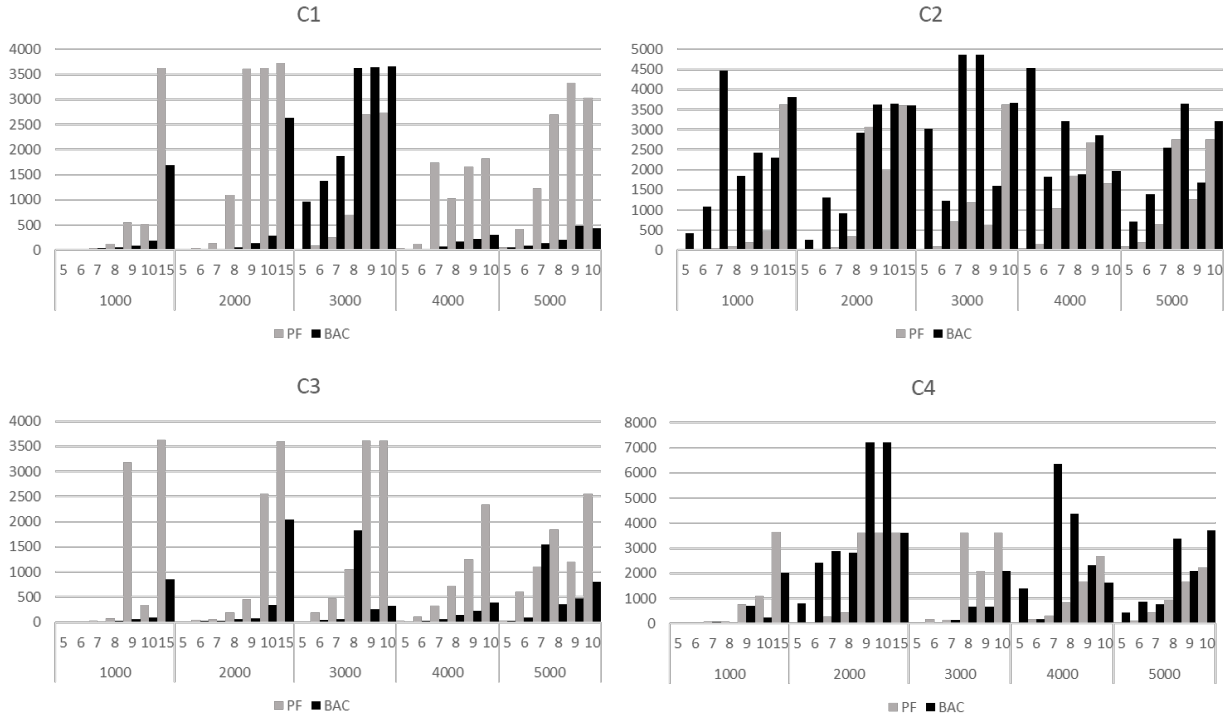


Figure A.17: Solving time of PF and BAC for $|S| = 100$, Low cost

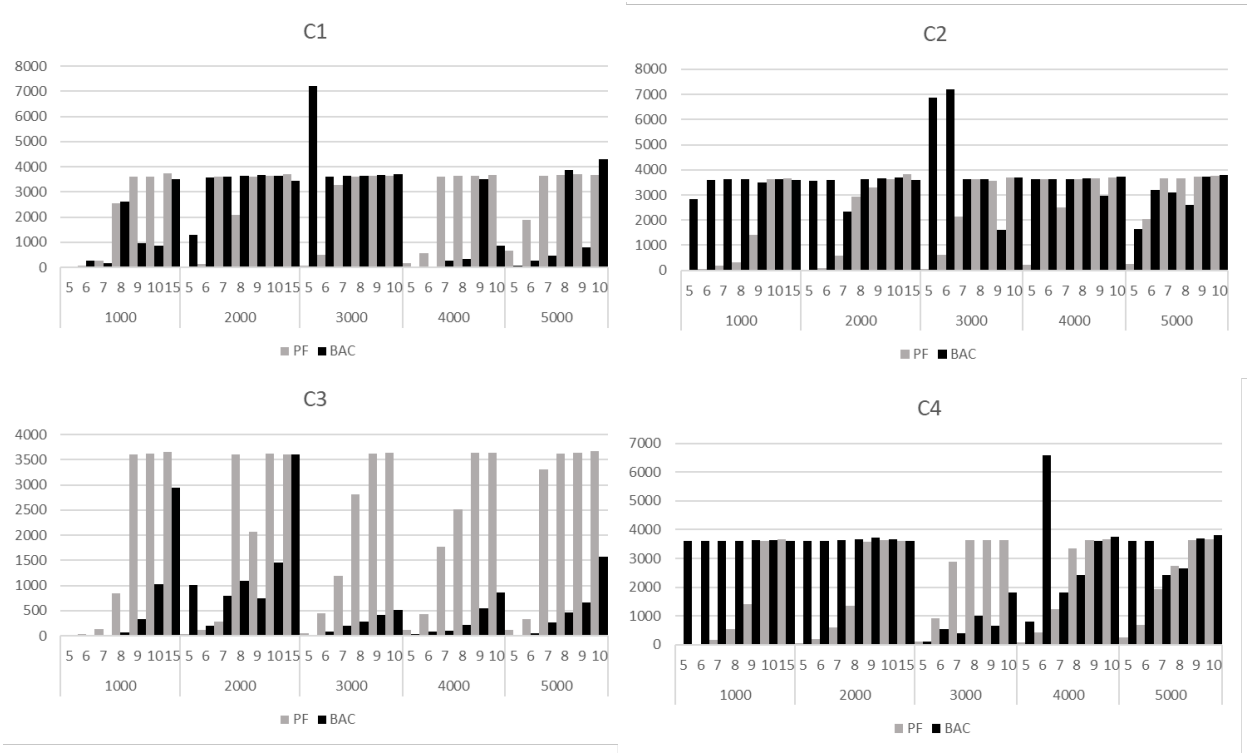


Figure A.18: Solving time of PF and BAC for $|S| = 200$, High cost

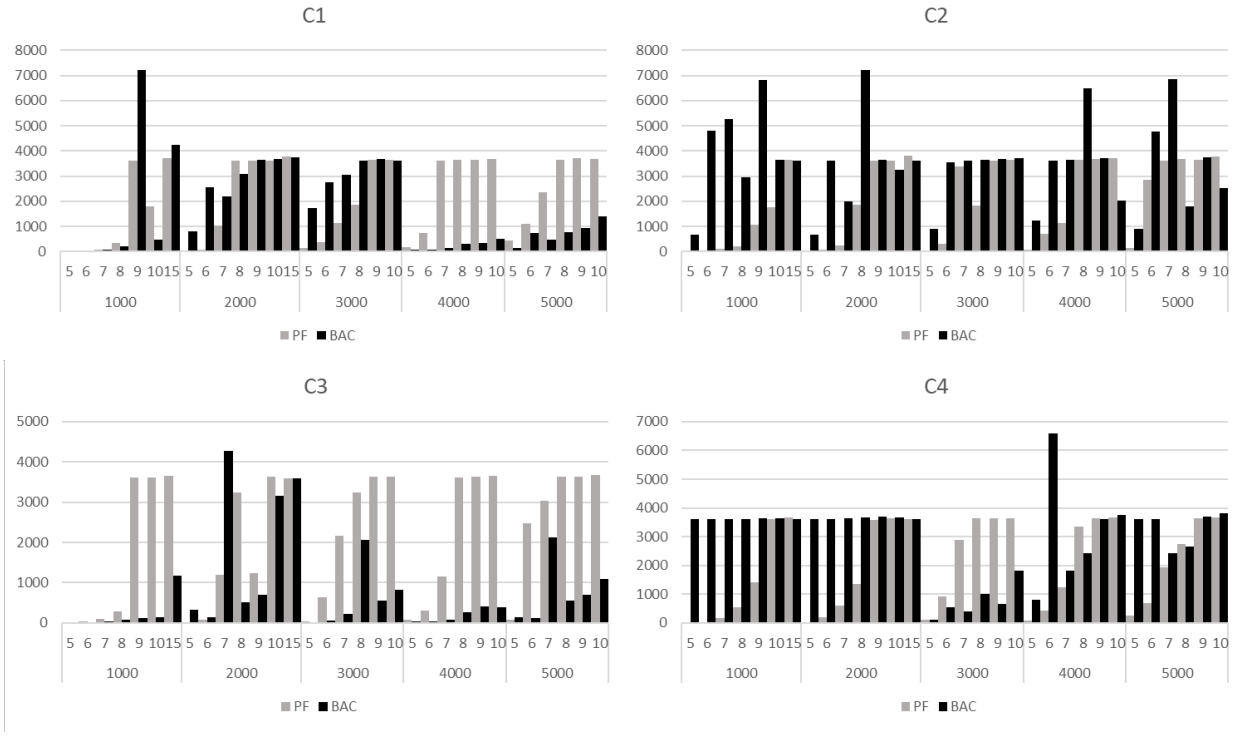


Figure A.19: Solving time of PF and BAC for $|S| = 200$, Low cost